

Aplicación de los conocimientos de Programación Orientada a Objetos para la construcción del aplicativo gestor de personas “Seatora”

Camilo Sanchez Cuestas, Adames Pardo Juan Camilo

Este documento tiene como objetivo exponer el camino y el proceso que se llevó a cabo en el momento de la planeación, diseño, creación y ejecución de un aplicativo capaz de gestionar un grupo de interés, que para este ejercicio se reconoce como “Seatora”. Se hará un breve recorrido sobre los conceptos y conocimientos adquiridos dentro de la clase de Programación Orientada a Objetos (POO) y además, cómo desde la lógica del grupo hicimos uso de estos conceptos para construir un gestor de personas con interfaz gráfica.

This document aims to expose the path and the process that was carried out at the time of planning, design, creation and execution of an application capable of managing a group of interest, which for this exercise is recognized as "Seatora". A brief tour will be made about the concepts and knowledge acquired in the Object Oriented Programming (OOP) class and also, how from the logic of the group we made use of these concepts to build a people manager with a graphical interface.

INTRODUCCIÓN

Para tratar los temas del paradigma Orientado a Objetos en la programación, se optó por comenzar con un primer acercamiento al lenguaje que fue la base para la construcción del aplicativo que es presentado y soportado con este documento, pero antes de meterse en materia, es de suma importancia hablar con mucha brevedad acerca de Java.

Java nació a mitades de 1994, por un grupo de 4 personas que buscaban “la creación de funciones mediante este “idioma”, que posteriormente pudieran ser aplicadas en otros entornos virtuales compatibles sin necesidad de crear desde cero la misma codificación” [1]. Este lenguaje resultó ser tan exitoso al punto de ser muy requerido su dominio por parte de las empresas, especialmente, y relacionado con este proyecto, por ser capaz de brindar las herramientas al programador de construir entornos digitales que lleven consigo bibliotecas de Java, constructores, encapsulamiento, persistencia, creación de listas dinámicas, gestión de grupos de personas, creación y modificación de archivos planos, entre otros.

Por otro lado, herramientas como JavaFX nos abren la puerta a un sin fin de cosas que podemos inventar juntando esto con los conocimientos del lenguaje Java, ya que JavaFX

funciona para crear y aplicar el código hecho anteriormente a interfaces gráficas que podemos ejecutar como aplicaciones de internet, utilizando las mencionadas bibliotecas de Java para crear estas interfaces e interactuar de una mejor manera con cualquier usuario [2]. Dicho esto, el dominio del lenguaje Java junto con el uso de JavaFX, hace que sea posible este proyecto, hace que sea posible hacer la gestión de “Seatora”.

MARCO TEÓRICO

La base del aplicativo “Seatora” es el lenguaje compilado y a su vez interpretado JAVA [3], siendo este el medio de funcionalidades esenciales para el desarrollo de escritorio, pero antes de continuar, es de vital importancia reconocer qué es un lenguaje compilado y un lenguaje interpretado, además de exponer sus diferencias con el objetivo de facilitar el entendimiento del lector con base al tema:

- Lenguaje Compilado: Según Netinbag.com, “Un lenguaje compilado es un lenguaje de programación de computadora cuyo código fuente generalmente se compila o traduce al código de máquina para producir un programa ejecutable. Los compiladores son programas de utilidad creados para traducir una implementación específica de un lenguaje de programación en un archivo binario ejecutable que está diseñado para ejecutarse en un determinado sistema operativo. Este archivo ejecutable independiente se puede ejecutar en cualquier plataforma compatible sin la ayuda de otro programa y sin la necesidad de volver a compilarlo. Algunos lenguajes compilados comunes son Ada, C ++ y Fortran.” [4]. Como fue mencionado anteriormente, un lenguaje de programación compilado requiere obligatoriamente un traductor a código máquina, este suele ser un IDE o un editor de texto con las extensiones necesarias para abarcar dicha traducción de los datagramas a los muy conocidos “bytes”.
- Lenguaje Interpretado: Este tipo de lenguaje se caracteriza principalmente por su ejecución basada de función a función [5], además de que a diferencia del lenguaje compilado, este no pasa por un proceso de compilado sino que el mismo entorno de desarrollo lo traduce a tiempo real.

Una vez comprendido los temas anteriores, se destaca una pregunta primordial, ¿cuál es la principal diferencia entre estos y por qué es un factor diferencial importante a la hora de

desarrollar?; pues bien, la principal diferencia entre estos tipos de lenguajes es la ejecución, puesto que un lenguaje compilado está optimizado para el momento de la ejecución, aunque esto signifique una carga adicional para el programador. Por otro lado, un lenguaje interpretado está optimizado para hacerle la vida más fácil al programador, aunque eso signifique una carga adicional para la máquina [6], es también por eso que se vuelve un factor crucial dependiendo del proyecto a desarrollar, puesto que puede significar una entrega a tiempo o un retraso inevitable con las deadline.

PROCEDIMIENTO

Para la construcción del gestor de personas “Seatora”, se partió de la base del parcial de segundo corte el cual trabaja la funcionalidad del programa sin ningún tipo de GUI o similar (puede encontrar la construcción del segundo parcial en el primer enlace del apéndice), por este motivo el objetivo de este informe es sustentar y soportar la entrega del mismo gestor, pero ahora aplicando los conocimientos de JavaFX para su visualización y manejo.

Como primera instancia se recalca la importancia de las librerías necesarias para liberar las funcionalidades de JavaFX, estás se representan en cada clase de manera estándar con los siguientes imports:

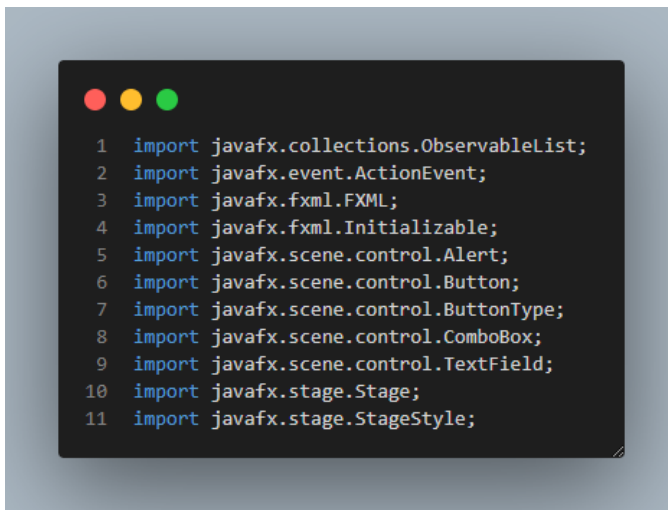


Figura 1. Imports necesarios para la ejecución de JavaFX

Una vez añadidos los imports, se procede a inicializar los ObservableList, estos son los “ArrayList” de JavaFX donde al inicializarse, se debe relacionar el modelo y el nombre que tendrá, quedando de la siguiente manera:

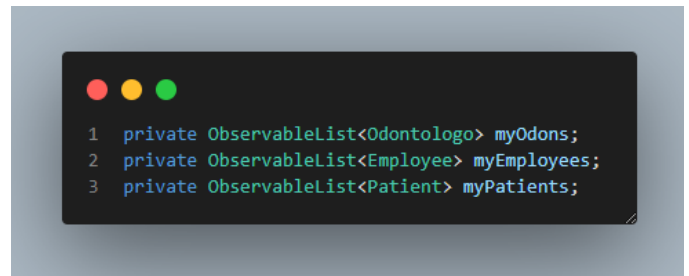


Figura 2. ObservableList JavaFX

Teniendo en cuenta la inicialización de los ObservableList, se construyeron una serie de 7 interfaces que cumplen el objetivo de navegación entre sí, donde cada una tiene su funcionalidad y objetivo, estas fueron hechas a partir del programa Scene Builder el cual es un software principal para las interfaces de JavaFX puesto que este permite un diseño manual y personalizado, desde la resolución, hasta los colores de cada ítem dentro del lienzo (llamado AnchorPane). Por otro lado, el gestor tuvo una organización de archivos y clases obedeciendo a las buenas prácticas de programación quedando de la siguiente manera:

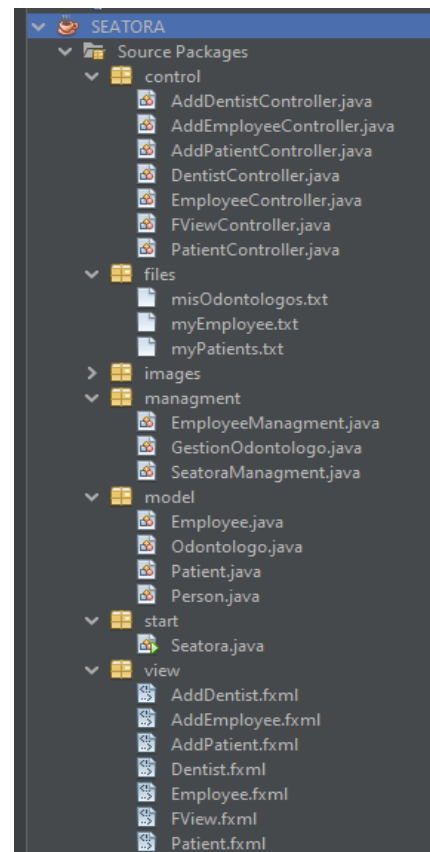
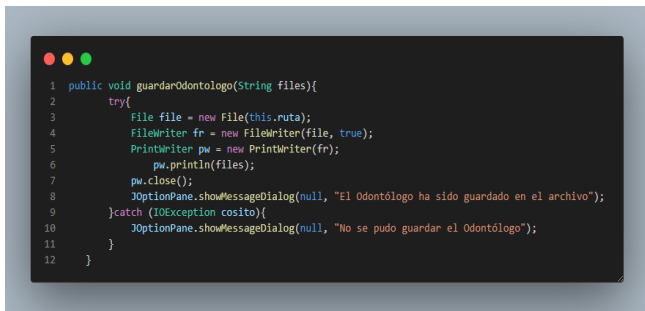


Figura 3. Organización de clases y archivos

Con lo que respecta a la persistencia, se resalta la presencia de operaciones CRUD dentro del proyecto, pero antes de continuar, con el objetivo de facilitar el entendimiento del lector, se definirá “persistencia”, así que a nivel de informática, la persistencia debe entenderse como la manera de que los datos sobrevivan [7], para llevar a cabo dicha

acción, se debe entender el funcionamiento de las memorias RAM (Random Access Memory), las cuales para no meterse en mucho en materia son las encargadas de almacenar datos de los programas que se están ejecutando en el momento, esto quiere decir que al dejar de ejecutarlos esta memoria perderá dicha información, por este motivo yace la idea de la persistencia de archivos, está ésta presente en diferentes formatos, como lo son; Persistencia en archivos planos (usada en este proyecto), persistencia en nube, persistencia en servidor local, entre otros [7].

Para entender la persistencia en nuestro proyecto, primeramente se debe definir el tipo de concurrencia usada, esta es la persistencia de archivos planos, la cual tiene el objetivo de convertir en String, los datos del programa para luego realizarles una separación por comas (llamado “split”), el cual se guardará automáticamente en archivos txt, esto dado al siguiente método dentro de cada gestión:



```

1 public void guardarOdontologo(String files){
2     try{
3         File file = new File(this.ruta);
4         FileWriter fw = new FileWriter(file, true);
5         PrintWriter pw = new PrintWriter(fw);
6         pw.println(files);
7         pw.close();
8         JOptionPane.showMessageDialog(null, "El Odontólogo ha sido guardado en el archivo");
9     } catch (IOException cosito){
10         JOptionPane.showMessageDialog(null, "No se pudo guardar el Odontólogo");
11     }
12 }

```

Figura 4. Método persistencia en txt

Con toda esta información, ya es posible la culminación del gestor de personas, este software se encuentra disponible en el segundo enlace del apéndice.

CONCLUSIÓN

En conclusión, se recalca la importancia de dominar un paradigma de programación capaz de abrir diversas puertas a un compendio de bastantes funcionalidades como lo es la Programación Orientada a Objetos en este caso ejecutado con el lenguaje compilado e interpretado: JAVA, a estos conocimientos se le debe sumar la capacidad de hacer interfaces gráficas con JavaFX, puesto que se pueden realizar proyectos con entornos digitales de gran calidad que pueden llegar a ser muy atractivos para las empresas y para los usuarios que día a día están buscando expertos que sean capaces de crear nuevas experiencias en las GUI que estén a la altura de las expectativas del mercado.

APÉNDICE

- Construcción del segundo parcial:
https://www.youtube.com/watch?v=R_Z0JxOxMX4&t=664s

- Repositorio de GitHub:
<https://github.com/juanadames1/SeatoraManagementwhitFX>

REFERENCES

- [1] “Historia de Java: un lenguaje de programación con recorrido - Tokio School,” *Tokio School*, Jul. 09, 2021.
<https://www.tokioschool.com/noticias/historia-java-el-origen-de-este-lenguaje-de-programacion/> (accessed May 26, 2022).
- [2] “¿Qué es JavaFX y para qué se utiliza? - Tokio School,” *Tokio School*, Dec. 16, 2019.
<https://www.tokioschool.com/noticias/que-es-javafx-usos/> (accessed May 26, 2022).
- [3] “¿Es el lenguaje Java un lenguaje compilado o interpretado? (Proceso de ejecución del programa Java) - programador clic,” *Programmerclick.com*, 2020.
<https://programmerclick.com/article/90001254966/> (accessed May 27, 2022).
- [4] “¿Qué es un lenguaje compilado?,” *Netinbag.com*, 2022.
<https://www.netinbag.com/es/internet/what-is-a-compiled-language.html> (accessed May 27, 2022).
- [5] admin, “Un lenguaje de programación interpretado es aquel que el código fuente se ejecuta directamente, instrucción a instrucción. Es decir, el Leer más,” *Lenguajes de programación*, Feb. 13, 2018.
<https://lenguajesdeprogramacion.net/diccionario/que-es-un-lenguaje-interpretado/> (accessed May 27, 2022).
- [6] G. Escobar, “Lenguajes compilados e interpretados,” *El Blog de Make it Real*, Nov. 06, 2017.
<https://blog.makeitreal.camp/lenguajes-compilados-e-interpretados/#:~:text=En%20general%2C%20un%20lenguaje%20compilado,carga%20adicional%20para%20la%20m%C3%A1quina.> (accessed May 27, 2022).
- [7] “Styde Limited,” *Styde.net*, 2015.
<https://styde.net/concurrencia-y-persistencia-en-programacion-orientada-a-objetos/> (accessed May 27, 2022).