|  |  |  |
| --- | --- | --- |
|  | **Carátula para entrega de prácticas** | |
| Facultad de Ingeniería | | Laboratorio de docencia |

[[1]](#footnote-1)

Laboratorios de computación

salas A y B

|  |  |
| --- | --- |
| *Profesor:* | Ing. José Antonio Ayala Barbosa |
| *Asignatura:* | Programación Orientada a Objetos |
| *Grupo:* | 1 |
| *No de Práctica(s):* | 11 |
| *Integrante(s):* | Mejia Valdiviezo Eduardo Javier 317078585  lalo.ejmv@gmail.com |
| *No. de Lista :* | 19  37 |
| *Semestre:* | 2021-1 |
| *Fecha de entrega:* | 14 / Enero / 2021 |
| *Observaciones:* |  |
|  |  |

CALIFICACIÓN: \_\_\_\_\_\_\_\_\_\_

Este documento contiene las actividades realizadas el 07 / 01/ 2021, en la clase de laboratorio de Programación Orientada a Objetos, relacionado al manejo de archivos.

# Introducción

# 

Objetivo: Implementar el intercambio de datos (lectura y escritura) entre fuentes externas (archivos y/o entrada y salida estándar) y un programa (en un lenguaje orientado a objetos).

En esta practica 1, veremos la importancia y la funcionalidad de crear, leer y escribir en archivos de texto plano.

Los programas necesitan comunicarse con su entorno, tanto para obtener datos e información que deben procesar, como para devolver los resultados obtenidos. El manejo de archivos se realiza a través de streams o flujos de datos desde una fuente hacia un repositorio. La fuente inicia el flujo de datos, por lo tanto, se conoce como flujo de datos de entrada. El repositorio termina el flujo de datos, por lo tanto, se conoce como flujo de datos de salida. Es decir, tanto la fuente como el repositorio son nodos de flujos de datos.

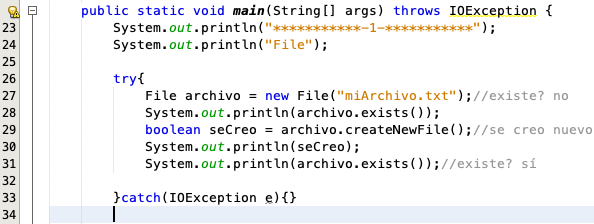
# Desarrollo

A continuación, se muestra las actividades, así cómo los procesos y los programas realizados, conteniendo cada uno, una breve explicación, composición y ejecución.

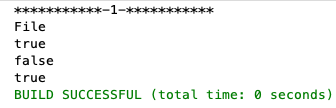
## Crear archivos de texto plano

En esta parte encontramos un poco sobre cómo crear estos tipos de archivos de texto planno y en dónde se encuentran, en este caso se llama “miArchivo.txt”, así cómo se muestra a continuación:

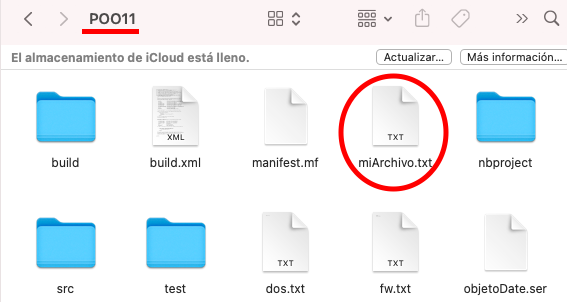
Las preguntas de los comentarios hechos en el código tendrán respuesta al momento de la ejecución.



Aquí podemos observar, ¿existe? True, ¿se creó nuevo? False y ¿existe? True.

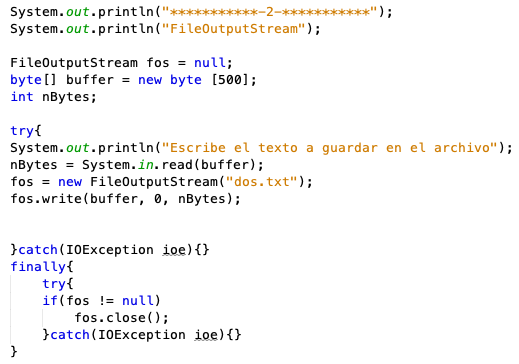


Para corroborar que se creó correctamente, únicamente tendremos que buscarlo en la carpeta de nuestro proyecto llamado “POO11” y ahí encontraremos “miArchivo.txt”.



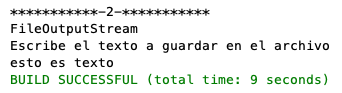
## Escribir en archivos de texto plano (FileOutputStream)

Aquí podemos encontrar cómo poder leer los archivos, primero un “FileOutputStream” que permite crar y escribir un flujo de bytes en un archivo de texto, mismo que se iguala a “null” y a contuniación un buffer donde se puede guardar cualquier cosa.

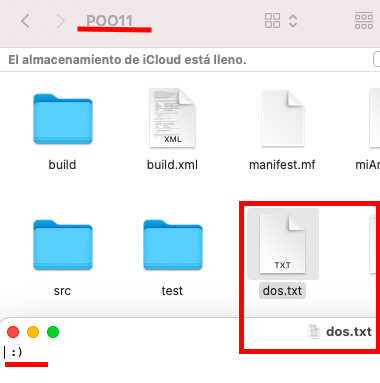


Y al ejecutar el programa se nos muestra lo siguiente:

En esta parte podremos escribir lo que sea y se guardará en nuestro archivo “dos.txt”.

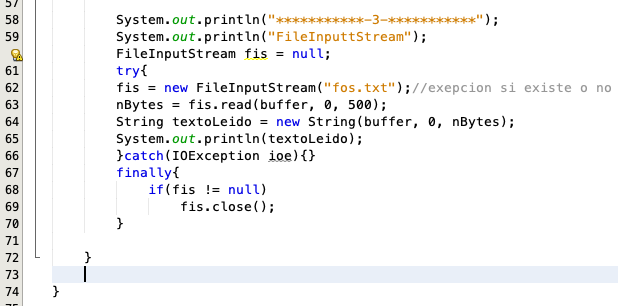


En nuestra carpeta del proyecto se encuentra nuestro archivo “dos.txt” y en este lo que hayamos guardado.

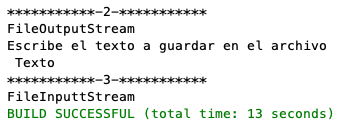


## Leer archivos de texto plano (FileInputStream)

Aquí lo que se realizó fue un código para poder leer lo escrito en nuestro archivo.

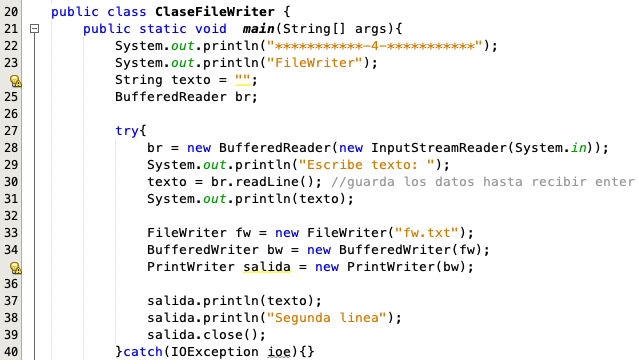


De igual forma se nos despliega en nuestra ejecución, la parte para escribir en nuestro archivo y cómo se ve, se ha leído lo que escribimos.

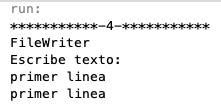


## Escribir caracteres en archivo (FileWriter)

Se creó otro paquete para continuar con las actividades, en este caso siguió “FileWriter” que nos sirve para escribir caracteres en nuestros archivos, en este caso se llama “fw.txt”.



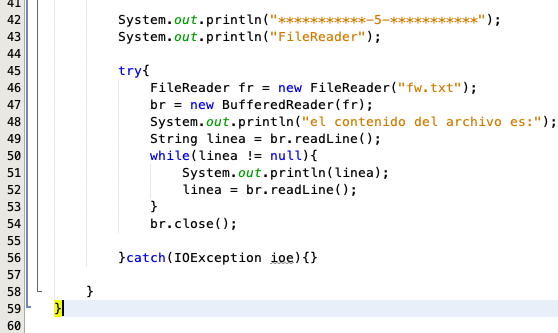
A continuación se nos muestra la ejecución y el contenido del archivo “fw.txt”.

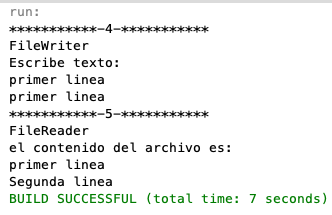




## Leer archivos (FileReader)

En esta parte se agregó “el contenido del archivo es:” para que posteriormente se muestre lo escrito y en la ejecución se despliega el mismo menú anterior.



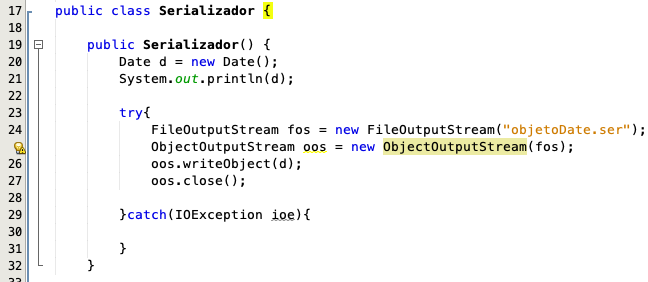


## Serializar Fecha

Esta parte es la más extensa ya que consta de varias partes, mismas que se mostrarán a continuación.

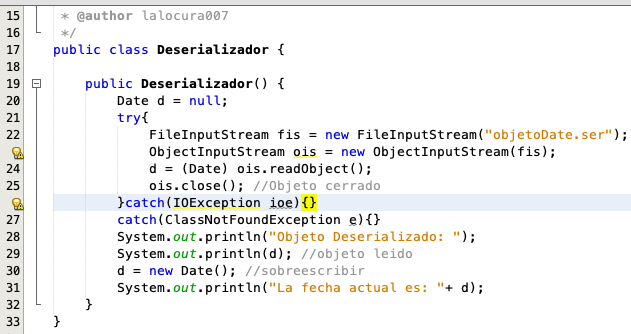
* Serializador

La serialización no necesita métodos por lo que es muy fácil usarala y nos sirve específicamente para convertir un objeto en bytes y se codificó de la siguiente manera:



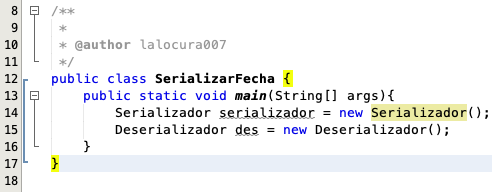
* Deserializador

Por el otro lado, tenemos esta parte que nos sirve para transformar los bytes en un objeto, que es igual de útil.

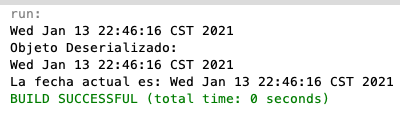


* SerializarFecha

En esta parte hacemos uso de las dos partes anteriores, primero serializamos y después deserializamos.



Por último, tenemos la ejecución, donde se nos muestra la fechas serializada y deserializada.



# Conclusión

El manejo de archivos resulta ser muy importante, ya que es una clase que nos permite escribir datos de tipo caracteres en un archivo. Este archivo al es un objeto en una computadora que puede almacenar informaciónel cual puede ser manipulado y de esta forma resulta ser muy sencillo.

# Referencias

<http://profesores.fi-b.unam.mx/annkym/LAB/poo_p11.pdf>

1. [↑](#footnote-ref-1)