



ESCUELA DE
INGENIERÍA EN CIENCIAS Y SISTEMAS
FACULTAD DE INGENIERÍA
UNIVERSIDAD DE SAN CARLOS DE GUATEMALA



Día, Fecha:

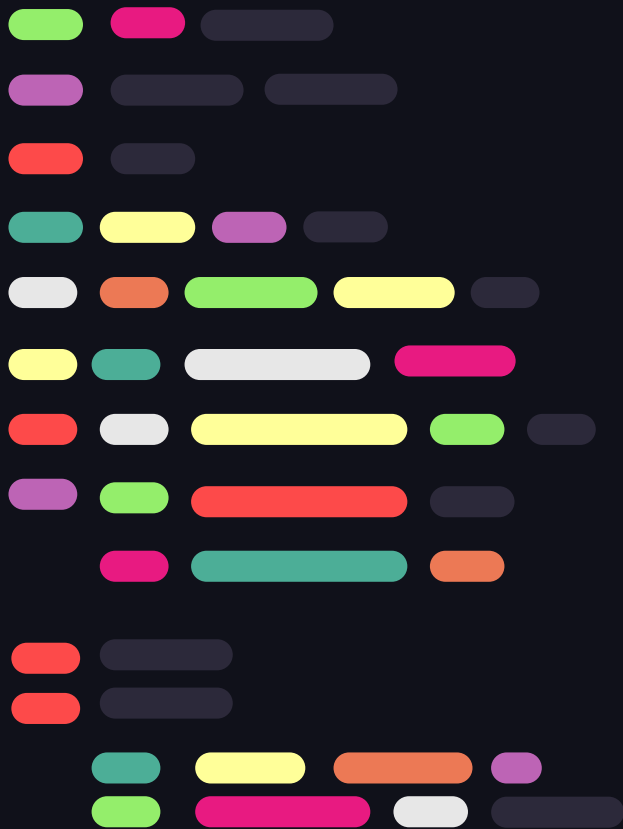
Miércoles, 12/06/2024

Hora de inicio:

13:00

Introducción a la Programación y Computación 1 [A]

Josué Rodolfo Morales Castillo



{ ..



Clase 8- Agenda

- Foro No. 2
- Preguntas práctica 1
- Horarios de Calificación
- Lectura hoja de calificación
- Manejo de Archivos y Serialización
- Crear .jar
 - Crear carpeta Archivos en la dist
- Verificar el .gitignore

} ..

Archivos de texto plano

Los archivos de texto plano son una forma fundamental de almacenar y manipular datos en el desarrollo de software. En Java, trabajar con archivos de texto es una tarea común y esencial para una amplia gama de aplicaciones.





Creación de Archivos (FileWriter)

```
1 import java.io.FileWriter;
2 import java.io.IOException;
3
4 public class EscribirArchivo {
5     public static void main(String[] args) {
6         try {
7             FileWriter archivo = new FileWriter("archivo.txt");
8             archivo.write(";Hola, mundo!");
9             archivo.close();
10            System.out.println("Texto escrito correctamente.");
11        } catch (IOException e) {
12            System.out.println("Error al escribir en el archivo.");
13            e.printStackTrace();
14        }
15    }
16 }
```

Lectura de Archivos(FileReader)

```
1 import java.io.FileReader;
2 import java.io.IOException;
3
4 public class LeerArchivo {
5     public static void main(String[] args) {
6         try {
7             FileReader archivo = new FileReader("archivo.txt");
8             int caracter;
9             while ((caracter = archivo.read()) != -1) {
10                 System.out.print((char) caracter);
11             }
12             archivo.close();
13         } catch (IOException e) {
14             System.out.println("Error al leer el archivo.");
15             e.printStackTrace();
16         }
17     }
18 }
```

Eliminación de Archivos(delete)

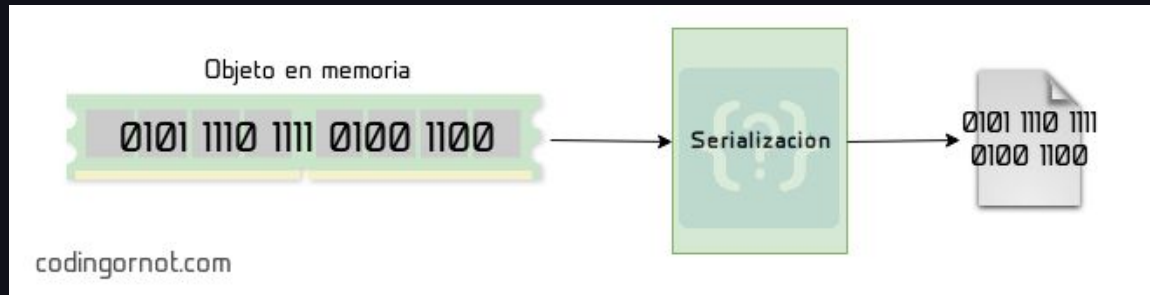
```
1  import java.io.File;
2
3  public class EliminarArchivo {
4      public static void main(String[] args) {
5          File archivo = new File("archivo.txt");
6          if (archivo.delete()) {
7              System.out.println("Archivo eliminado correctamente.");
8          } else {
9              System.out.println("Error al eliminar el archivo.");
10         }
11     }
12 }
13
```

Modificar archivo

```
1  String nombreArchivo = "archivo.txt";
2
3  try {
4      // Abrir el archivo en modo de lectura y escritura
5      FileWriter fileWriter = new FileWriter(nombreArchivo, true);
6      BufferedWriter bufferedWriter = new BufferedWriter(fileWriter);
7      PrintWriter printWriter = new PrintWriter(bufferedWriter);
8
9      // Modificar el contenido del archivo
10     printWriter.println("Nueva línea agregada al archivo.");
11
12     // Cerrar el archivo
13     printWriter.close();
14
15     System.out.println("El archivo se ha modificado correctamente.");
16 } catch (IOException e) {
17     System.out.println("Ocurrió un error al modificar el archivo: " + e.getMessage());
18 }
```

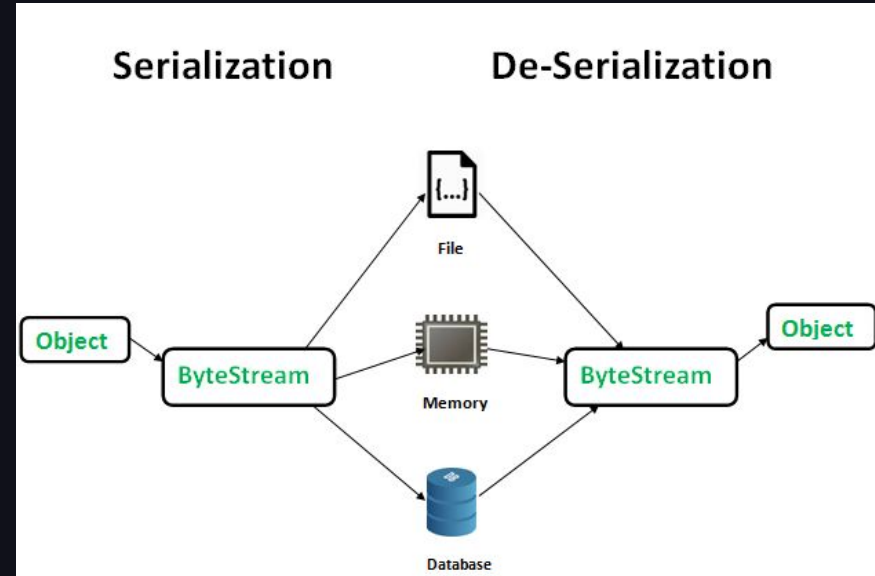
Serialización de objetos

La serialización es el proceso de convertir un objeto en una secuencia de bytes que puede ser almacenada, transmitida o guardada en un archivo. Para que un objeto sea serializable, su clase debe implementar la interfaz **Serializable**. Una vez que un objeto es serializado, puede ser guardado o transmitido utilizando clases como **ObjectOutputStream**.



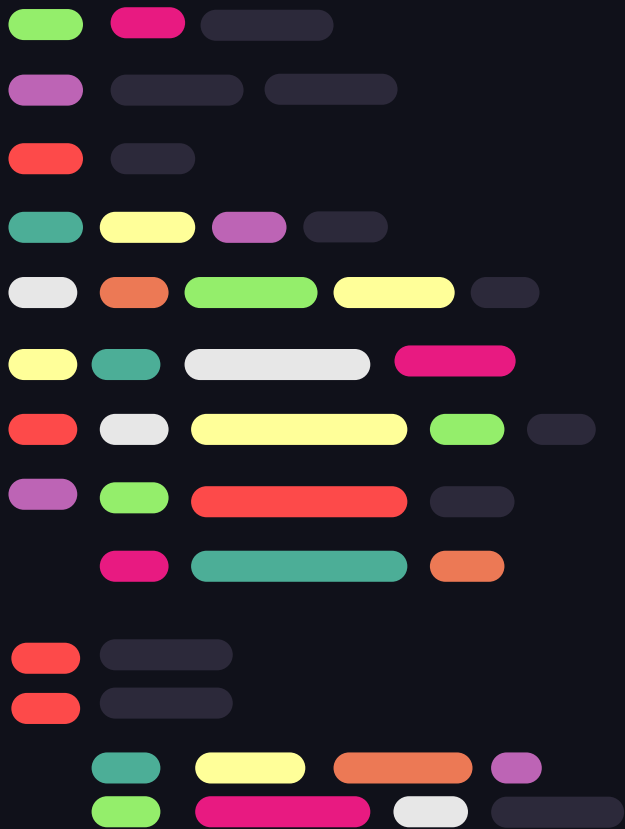
Deserialización de objetos

Es el proceso inverso de la serialización. Convierte una secuencia de bytes (que representa un objeto serializado) de vuelta en un objeto. Esto se hace principalmente para reconstruir objetos previamente serializados, por ejemplo, cuando se lee un objeto de un archivo. Para deserializar un objeto, se utiliza la clase `ObjectInputStream`.



¿Dudas?





{ ..



Ejemplo

} ..