Antes de empezar descarga el código fuente asociado a esta tarea e impórtalo en Eclipse (File->Import->Existing Java Project).

1. Identifica las diferentes responsabilidades que existen en este programa. Haz una lista con ellas.

-Interactuar con el usuario para pedir nombres de ficheros

-Transformar el contenido del fichero(en el caso inicial a XML)

-Llevar el flujo principal del programa

1. Refactoriza la aplicación para cumplir estos **objetivos**:
   1. El origen de datos (ahora ficheros) pueda ser distinto **y/o** el destino de los datos también.
   2. La transformación que se hace de la entrada pueda ser otra representación basada en texto cualquiera (no a XML).

Una vez refactorizada,

1. Responde a estas cuestiones:
   * ¿**Qué** principio o principios SOLID has empleado para cada uno de los dos objetivos anteriores, **cómo** los has empleado y **por qué**?

Utilizo SRP para identificar las clases mínimas que debo crear según la responsabilidad que tienen.

Utilizo OCP para identificar que clases cerrar y donde dejar una interfaz punto de expansión. En concreto toXML pasa a ser una clase cerrada a modificación con un punto de expansión en la interfaz que implementa, de forma análoga se actúa con la responsabilidad de “origen de ficheros”.

Se utiliza el principio de sustitución de Liskov al permitir substituir las clases que llama el main por cualquiera de sus subclases o implementaciones.

La solución propuesta no cumple al completo con el principio de segregación de interfaces porque agrupo las responsabilidades de pedir fichero de entrada y fichero de salida en una única responsabilidad. Queda una interfaz para ambas acciones, si solo quieres hacer una acción la interfaz te obliga a implementar ambas.

Al crear dependencias en interfaces y no en clases concretas estamos aplicando el principio de inversión de dependencias.

1. Elabora una tabla para documentar las responsabilidades de las clases

|  |  |
| --- | --- |
| Clase | Responsabilidad |
| ConverterApp | Flujo Principal Del programa |
| GetterFileConsoleInput | Generar objetos File a partir de la entrada por consola |
| ToXML | Escribir en el fichero salida la entrada convertida a XML |

1. Elabora una tabla para documentar el principio OCP (Open Closed Principle) en tu código. Una clase que respeta el principio OCP, está cerrada para modificación pero abierta para extensión, siempre centrándonos en una modificación futura concreta. Por lo tanto, la tabla debe contener estas columnas.

|  |  |  |
| --- | --- | --- |
| Clase | Modificación posible | Punto de extensión |
| FormatterToXML | Convertir a otro formato | FormatterInterface |
| GetterFileConsoleInput | Entrada por parámetros del main en vez de consola | GetterFileInteface |

1. Modifica la aplicación para que la salida se produzca por pantalla y no a fichero. ¿Tuviste que cambiar código existente a mayores que el método *main*? Describe brevemente la modificación.