Este programa en Java está diseñado para trabajar con la API pública de Star Wars (SWAPI). Permite descargar información en formato JSON, convertirla a XML, guardar datos en ficheros y mostrarlos por pantalla. El objetivo es practicar el manejo de ficheros, objetos, JSON, XML y peticiones HTTP.

El proyecto está hecho con Maven y usa la librería Gson para procesar JSON. Para trabajar con XML se utiliza la API DOM incluida en Java. También se usan clases de Java para serializar objetos y guardarlos en disco.

El programa principal muestra un menú con las siguientes opciones:

1. Conversor (film JSON -> resultado.xml)
2. Añadir Personaje
3. Salvar Personajes (serialización)
4. Especie del Personaje
5. Mostrar datos XML (resultado.xml)
6. Salir

A continuación se explica qué hace cada opción.

Opción 1: Conversor
El usuario introduce el código de una película, por ejemplo 1. El programa hace una petición HTTP a la dirección https://swapi.dev/api/films/1/?format=json y recibe un archivo JSON con la información de la película. Después convierte ese JSON a formato XML usando la clase JsonToXmlConverter. Se crean etiquetas XML para los campos principales (título, director, fecha, personajes, planetas, etc.) y se guarda el resultado en un archivo llamado resultado.xml. Este archivo se puede abrir con un navegador o editor de texto para comprobar su estructura.

Opción 2: Añadir Personaje
El usuario introduce un código de personaje, por ejemplo 1. El programa accede a https://swapi.dev/api/people/1/?format=json y descarga la información del personaje en formato JSON. Luego crea un objeto Personaje con los datos principales (nombre, altura, peso, color de pelo, color de piel, color de ojos, año de nacimiento, género y lista de especies). Antes de añadirlo a la lista de personajes en memoria, se comprueba que no exista ya un personaje con el mismo nombre. Si no existe, se añade al ArrayList de personajes.

Opción 3: Salvar Personajes
Esta opción guarda todos los personajes del ArrayList en un archivo llamado personajes.dat. El guardado se realiza mediante serialización, usando ObjectOutputStream.

De este modo, los personajes se pueden recuperar en futuras ejecuciones del programa sin tener que volver a descargarlos.

Opción 4: Especie del Personaje
El usuario introduce un código de personaje. El programa obtiene el JSON de ese personaje y revisa el campo species, que contiene una lista de URLs con las especies del personaje. Por cada URL, hace una nueva petición HTTP y obtiene el nombre de la especie. Finalmente, muestra las especies por pantalla. Por ejemplo, si el personaje es Luke Skywalker, la especie mostrada será Human.

Opción 5: Mostrar datos XML
Esta opción lee el archivo resultado.xml generado por la opción 1. Usa un parser DOM para analizar el contenido del XML y muestra por pantalla los nombres de las etiquetas y sus valores. Así se pueden ver los datos principales de la película sin abrir el archivo manualmente.

Opción 6: Salir
Finaliza el programa. Se recomienda antes usar la opción 3 para guardar los personajes en el fichero.

Cuando el programa se inicia, intenta cargar los personajes guardados en personajes.dat mediante deserialización. Si el archivo existe, se leen los objetos y se añaden al ArrayList de personajes para que el usuario pueda seguir trabajando con ellos.

El programa genera dos ficheros principales:
resultado.xml, que contiene los datos de una película en formato XML.
personajes.dat, que guarda los objetos de tipo Personaje en formato binario.

Ejemplo de ejecución:
El usuario elige la opción 1 e introduce el código 1. El programa crea el archivo resultado.xml.
Luego elige la opción 2 y añade el personaje con código 1 (Luke Skywalker).
Después selecciona la opción 3 para guardar los personajes.
Si elige la opción 4 con el código 1, el programa muestra que la especie del personaje es Human.
Con la opción 5 puede ver por pantalla los datos de la película guardada en XML.

En resumen, el programa combina varias técnicas de programación en Java: consumo de API REST mediante HTTP, manejo de JSON con Gson, creación y lectura de XML mediante DOM, uso de colecciones dinámicas (ArrayList), serialización de objetos y persistencia de datos entre ejecuciones. Además, evita duplicados en la lista de personajes y permite al usuario trabajar de forma interactiva desde un menú de consola.