Práctica 12
Archivos

(Rev 0.9)

1 Objetivos

Que el alumno aprenda a utilizar archivos para tareas comunes en la programación, como la creación de paginas web del lado del servidor, y la serialización de objetos.

2 Pre-laboratorio

2.1 Cuestionario

(Las preguntas de los cuestionarios vendrán en sus exámenes parciales del salón.)

1. ¿Qué es la serialización y para qué se usa?
2. ¿Qué son los lenguajes de etiquetas?
3. ¿Qué son y para qué se usan los archivos con formato xml?

2.2 Actividades

1.

3 Teoría

3.1 Serialización

La serialización es la acción de convertir a un objeto en un flujo de bytes para guardarlo en un archivo o transmitirlo por algún canal de comunicación. La operación contraria, deserialización, toma un flujo de bytes y lo convierte en un objeto. A primera vista parece que es lo mismo que únicamente guardar en un archivo los atributos del objeto, pero la serialización es algo más complicado que eso. Si uno de los atributos es una referencia o apuntador a otro objeto, entonces hemos de seguir a la referencia hasta el objeto real, porque de nada nos va a servir almacenar nada más la dirección (el apuntador). Piense en los siguientes escenarios:

- ¿Cómo guardaría Ud. una lista enlazada con n nodos en un archivo, para luego recuperlos otra vez como lista enlazada?
- ¿Y si en lugar de una lista enlazada se tratara de un árbol o un grafo lo que Ud. quiere guardar?

En ninguno de los tres casos anteriores puede guardar las direcciones en memoria de los nodos, no le servirían de nada la siguiente vez que carga el programa; tiene que guardar la información real contenida en cada uno de los nodos.

Además de la situación recién descrita se presenta otro problema a resolver: las facilidades del lenguaje para guardar o leer información de un archivo se limitan a los tipos básicos, incluyendo las cadenas std::string. ¿Qué hay que hacer entonces para poder guardar o leer atributos que sean de tipos compuestos – recordando que las clases son tipos compuestos – ? Cada clase habría que hacerla serializable, es decir, que las instancias de estas clases deberán saber cómo guardarse en un archivo. Para esto hemos de tener una interfaz ISerializable que cada objeto deberá implementar.

Lenguajes como Java y C# incluyen *de-facto* esta característica, pero C++ no. En términos prácticos esto significa que Java o C# es más fácil serializar y deserializar que con C++, pero aún así en estos tres lenguajes hay que "enseñarle" al compilador a serializar tipos compuestos. Como parte del lenguaje en Java y C# existe una interfaz ISerializable que únicamente hay que implementar. En C++ hemos también, además de llevar a cabo tal implementación, definir la interfaz.

TBD(En la Actividad 3 vamos a ver un ejemplo de serialización en C++.)

4 Actividades

Actividad 1: Guardando y extrayendo los datos de objetos Alumno

En esta actividad Ud. se familiarizará con el uso de los archivos de texto.

1. Cree una clase Alumno (o use una que ya tenga) y guarde en un archivo los atributos nombre, promedio y calificación. Tenga en cuenta que esta operación de guardar NO pertenece a la clase, hágalo directamente en el driver program. Para esto su clase deberá implementar los setters/getters para cada atributo (vea el siguiente punto).
2. En el mismo driver program lea los datos del archivo y guárdelos en una nueva instancia de Alumno. Agregue una operación Imprimir() a la clase que imprima los tres atributos.

Actividad 2: Generando una página web

En esta actividad Ud. va a programar una pequeña aplicación que leerá algunos datos por parte del usuario, y a continuación generará una página web muy simple, emulando parte de las tareas que hacen los servidores web.

3. Termine la implementación de las clases que se indica en el código fuente anexo. *TIP: Escriba la implementación de una clase y prúebela. Luego continúe con la siguiente clase, y así sucesivamente*.
4. En el driver program pida al usuario los datos que se necesitan y genere la página web.
5. Abra el archivo en su navegador y comente los resultados.

Actividad 3: Serializando un objeto

(Pendiente)