

# Clase 24: Diccionarios IV

El problema de todas las estructuras vistas hasta ahora es la falta de persistencia. Si la aplicación termina, los datos se pierden.

Una solución consiste en guardar los datos en un archivo de acceso aleatorio (random access file). Un archivo de acceso directo puede visualizarse como un arreglo que se guarda en el disco duro. Los elementos de este arreglo, son del tipo byte.

Un byte es un entero cuyo rango es reducido y es la unidad básica para representar otros tipos en computación. Por ejemplo, una letra se suele representar usando un byte; un int, por cuatro y un String, por un byte por cada caracter más uno.

Para usar estos archivos, Java provee la clase `RandomAccessFile`

## 1 Apertura de un archivo

Para abrir un archivo, el código es

```
RandomAccessFile r=new RandomAccessFile("nombreArchivo","rw");
```

El segundo argumento indica si el archivo se está abriendo para leerlo ("r") o para leerlo y escribirlo ("rw").

## 2 Escritura en el archivo

En el archivo se puede escribir usando el método `writeInt` o `writeBytes`

```
r.writeInt(5);  
r.writeBytes("ejemplo");
```

El archivo que acabamos de crear, usa 12 bytes (4 para el entero y 8 para el String).

## 3 Cerrando un archivo

El archivo se cierra usando

```
r.close();
```

## 4 Lectura del archivo

Si el archivo acaba de abrirse para lectura, se pueden leer los datos que acabamos de escribir:

```
int i=r.readInt();  
String s=r.readLine();
```

El orden si importa! Si el código anterior fuera

```
String s=r.readLine();  
int i=r.readInt();
```

en las variables quedaría lo que en computación se llama basura, pues Java trataría de interpretar un entero como un String y viceversa. Los resultados son muy difíciles de prever.

A diferencia de los archivos de texto, para poder leer estos archivos se debe conocer su estructura ("formato"). Esta es la razón por la cual si se trata de abrir un documento Excel con Autocad, el computador lee cualquier cosa.

## 5 seek

El verdadero poder de estos archivos y lo que le da el nombre de "aleatorio" es la capacidad de "saltar" a cualquier lado del archivo para leer o escribir. Esto los diferencia de los archivos de texto comunes, que son secuenciales, donde solo se puede escribir al final o leer la siguiente línea.

El método para saltar a cualquier parte del archivo se llama `seek`.

El código de la sección 4 podría escribirse también de esta forma:

```
r.seek(4);  
String s=r.readLine();  
r.seek(0);  
int i=r.readInt();
```

Es más, si quisiéramos cambiar "ejemplo" por "example", el código es

```
r.seek(4);  
r.writeBytes("example");
```

## 6 length

El método `length` devuelve el largo del archivo (en bytes)

## 7 Problema resuelto

Vamos a suponer que la clase Diccionario ha sido implementada usando un RandomAccessFile

```
class Diccionario {
    RandomAccessFile datos;
    Diccionario() {
        datos=new RandomAccessFile("datos","rw");
    }
}
```

El formato del archivo es

```
llave1 (30 caracteres)
valor1 (80 caracteres)
llave2 (30 caracteres)
valor2 (80 caracteres)
...
```

Esto implica que la llave  $i$  comienza en la posición  $110*(i-1)$  y que el valor  $i$  comienza en la posición  $110*(i-1)+30$ .

Vamos a suponer, además, que el archivo está ordenado por la llave, es decir

```
llave(i)<llave(i+1)
```

Sabiendo esto, vamos a implementar el método get, usando búsqueda binaria

```
String get(String llave) {
    int ini=0;
    int fin=datos.length()/110-1;
    while (ini<fin) {
        int m=(ini+fin)/2;
        r.seek(m*110);
        String ll=r.readLine();
        if (ll.equals(llave))
            return r.readLine();
        else if (ll.compareTo(llave)<0)
            ini=m+1;
        else
            fin=m-1;
    }
    return null;
}
```

## 8 Rendimiento

Leer desde un archivo (o escribir) en forma secuencial es bastante eficiente. En un buen disco duro, es posible escribir varios millones de bytes por segundo. Sin embargo, hacer un seek toma del orden de milisegundos.

Por otro lado, escribir un byte toma el mismo tiempo que tomar un par de kilos. Dicho de otra forma, la menor cantidad de información que se puede escribir es al menos un par de miles de bytes.