Programación de Computadores: Java - Archivos y Excepciones

Juan F. Pérez

Departamento MACC
Matemáticas Aplicadas y Ciencias de la Computación
Universidad del Rosario

juanferna.perez@urosario.edu.co

Segundo Semestre de 2017

Contenidos

Archivos

Archivos

Variables almacenan información

Archivos

Variables almacenan información

Variables locales disponibles durante la ejecución del método

Archivos

Variables almacenan información

Variables locales disponibles durante la ejecución del método

Variables de instancia disponibles mientras el objeto esté vivo

Archivos

Variables almacenan información

Variables locales disponibles durante la ejecución del método

Variables de instancia disponibles mientras el objeto esté vivo

Todas las variables desaparecen al terminar el programa

Archivos

Variables almacenan información

Variables locales disponibles durante la ejecución del método

Variables de instancia disponibles mientras el objeto esté vivo

Todas las variables desaparecen al terminar el programa

¿Cómo mantener información disponible después de que el programa ha sido ejecutado?

Archivos

Variables almacenan información

Variables locales disponibles durante la ejecución del método

Variables de instancia disponibles mientras el objeto esté vivo

Todas las variables desaparecen al terminar el programa

¿Cómo mantener información disponible después de que el programa ha sido ejecutado?

Archivos

Archivos

Variables almacenan información

Variables locales disponibles durante la ejecución del método

Variables de instancia disponibles mientras el objeto esté vivo

Todas las variables desaparecen al terminar el programa

¿Cómo mantener información disponible después de que el programa ha sido ejecutado?

- Archivos
- Almacenados en el disco

Archivos

Variables almacenan información

Variables locales disponibles durante la ejecución del método

Variables de instancia disponibles mientras el objeto esté vivo

Todas las variables desaparecen al terminar el programa

¿Cómo mantener información disponible después de que el programa ha sido ejecutado?

- Archivos
- Almacenados en el disco
- Disponibles en el sistema de archivos del sistema operativo

Leyendo Archivos

Identificando archivos:

Dirección (path): ubicación en sistema de archivos (e.g., `C:\\misArchivos\\programacion`)

Nombre (e.g., `soledad`)

Extensión (e.g., `.txt`)

Leyendo Archivos

Identificando archivos:

Dirección (path): ubicación en sistema de archivos (e.g., `C:\\misArchivos\\programacion`)

Nombre (e.g., `soledad`)

Extensión (e.g., `.txt`)

Archivos de texto

Legibles por humanos

Se pueden acceder caracter a caracter o línea a línea (secuencialmente)

Leyendo Archivos en Java

Clase `FileReader`

Leyendo Archivos en Java

Clase `FileReader`

`http://docs.oracle.com/javase/8/docs/api/?java/io/FileReader.html`

Leyendo Archivos en Java

Clase FileReader

http://docs.oracle.com/javase/8/docs/api/?java/io/FileReader.html

```
String fullname = "D:\\\\java\\proyectoArchivos\\data\\
    pennylane.txt";
FileReader fReader = new FileReader(fullname);
int c = fReader.read();
if (c != -1){
 println((char)c);
}
c = fReader.read();
if (c != -1){
 println((char)c);
}
fReader.close();
```

Leyendo Archivos en Java

¿Qué pasa si el archivo no existe o no puede abrirse?:

Leyendo Archivos en Java

¿Qué pasa si el archivo no existe o no puede abrirse?:

Se genera un error o excepción

Leyendo Archivos en Java

¿Qué pasa si el archivo no existe o no puede abrirse?:

Se genera un error o excepción

Es un error que puede detectarse o manejarse

Leyendo Archivos en Java

¿Qué pasa si el archivo no existe o no puede abrirse?:

Se genera un error o excepción

Es un error que puede detectarse o manejarse

¿Cómo?

Excepciones

Excepciones

Algunas instrucciones pueden generar errores

Si esto se considera anticipadamente, se puede reaccionar

Excepciones

Algunas instrucciones pueden generar errores

Si esto se considera anticipadamente, se puede reaccionar

En Java

Excepciones

Algunas instrucciones pueden generar errores

Si esto se considera anticipadamente, se puede reaccionar

En Java

Instrucciones que generan errores *lanzan excepciones* (throw exception)

Si se detecta el error, se puede *atrapar* (catch) la excepción y ejecutar instrucciones para mitigar, recuperarse del error o avisar sobre el mismo

Bloque `try-catch`

Leyendo Archivos en Java con Excepciones

```
String fullname = "D:\\\\java\\proyectoArchivos\\data\\
    pennylane.txt";
try{
 FileReader fReader = new FileReader(fullname);
 int c = fReader.read();
 if (c != -1){
  println((char)c);
 }
 c = fReader.read();
 if (c != -1){
  println((char)c);
 }
 fReader.close();

}catch(Exception e){
 println("El archivo no pudo leerse");
}
```

Buffered Reader y más Excepciones

BufferedReader

`FileReader` solo permite leer un caracter a la vez.

BufferedReader

`FileReader` solo permite leer un caracter a la vez.

Introducimos `BufferedReader` que permite leer toda una línea

BufferedReader

`FileReader` solo permite leer un caracter a la vez.

Introducimos `BufferedReader` que permite leer toda una línea

Lee una línea completa y pasa a la siguiente

BufferedReader

`FileReader` solo permite leer un caracter a la vez.

Introducimos `BufferedReader` que permite leer toda una línea

Lee una línea completa y pasa a la siguiente

```
String fullname = "D:\\\\java\\proyectoArchivos\\data\\
    pennylane.txt";
try{
 FileReader fReader = new FileReader(fullname);
 BufferedReader bufReader = new BufferedReader(fReader);
 String linea = bufReader.readLine();
 println(linea);
 linea = bufReader.readLine();
 println(linea);
 bufReader.close();
}catch(Exception e){
 println("EL archivo no pudo leerse");
}
```

Leyendo todas las líneas con BufferedReader

```
String fullname = "D:\\\\java\\proyectoArchivos\\data\\
    pennylane.txt";

try{
 FileReader fReader = new FileReader(fullname);
 BufferedReader bufReader = new BufferedReader(fReader);
 String linea = bufReader.readLine();
 while(linea != null){
  println(linea);
  linea = bufReader.readLine();
 }
 bufReader.close();
}catch(Exception e){
 println("EL archivo no pudo leerse");
}
```

Excepciones Específicas

```
String fullname = "D:\\\\java\\proyectoArchivos\\data\\
    pennylane.txt";
try{
 FileReader fReader = new FileReader(fullname);
 BufferedReader bufReader = new BufferedReader(fReader);
 String linea = bufReader.readLine();
 while(linea != null){
  println(linea);
  linea = bufReader.readLine();
 }
 bufReader.close();

}catch(FileNotFoundException e){
 println("EL archivo no se encontró");
}catch(IOException e){
 println("EL archivo pudo leerse");
}
```

Escribiendo Archivos

Escribiendo Archivos en Java

`FileWriter`: permite escribir en un archivo caracter por caracter

Escribiendo Archivos en Java

`FileWriter`: permite escribir en un archivo caracter por caracter

`PrintWriter` permite escribir línea por línea usando un `FileWriter`

Escribiendo Archivos en Java

`FileWriter`: permite escribir en un archivo caracter por caracter

`PrintWriter` permite escribir línea por línea usando un `FileWriter`

También puede generar un excepción.

Escribiendo Archivos en Java

`FileWriter`: permite escribir en un archivo caracter por caracter

`PrintWriter` permite escribir línea por línea usando un `FileWriter`

También puede generar un excepción.

```
String fullname = "D:\\\\java\\proyectoArchivos\\data\\
    hola.txt";
try{
 FileWriter fWriter = new FileWriter(fullname);
 PrintWriter writer = new PrintWriter(fWriter);
 writer.println("hola");
 writer.println("este es mi primer archivo");
 writer.println("saludos,");
 writer.println("yo");
 writer.close();
}catch(IOException e){
 println("EL archivo pudo escribirse");
}
```

Otro Ejemplo Escribiendo Archivos en Java

```
String fullname = "D:\\\\java\\proyectoArchivos\\data\\
    abecedario.txt";
try{
 FileWriter fWriter = new FileWriter(fullname);
 PrintWriter writer = new PrintWriter(fWriter);
 char c = 'A';
 for(int i= 0; i < 26; i++){
  writer.println((char)(c+i));
 }
 writer.close();
}catch(IOException e){
 println("EL archivo pudo escribirse");
}
```

Ubicando Archivos con una Interfaz Gráfica

Interfaz Gráfica para *Abrir* Archivos

Método familiar de seleccionar archivos (sistema operativo)

Interfaz Gráfica para *Abrir* Archivos

Método familiar de seleccionar archivos (sistema operativo)

Evita escribir direcciones completas (errores)

Interfaz Gráfica para *Abrir* Archivos

Método familiar de seleccionar archivos (sistema operativo)

Evita escribir direcciones completas (errores)

Clase `JFileChooser`

Interfaz Gráfica para *Abrir* Archivos

Método familiar de seleccionar archivos (sistema operativo)

Evita escribir direcciones completas (errores)

Clase `JFileChooser`

Cuadro de diálogo para *abrir* un archivo: `showOpenDialog()`

Interfaz Gráfica para *Abrir* Archivos

Método familiar de seleccionar archivos (sistema operativo)

Evita escribir direcciones completas (errores)

Clase `JFileChooser`

- Cuadro de diálogo para *abrir* un archivo: `showOpenDialog()`
- Botón seleccionado (Save/Cancel): `JFileChooser.APPROVE_OPTION`

Interfaz Gráfica para *Abrir* Archivos

Método familiar de seleccionar archivos (sistema operativo)

Evita escribir direcciones completas (errores)

Clase `JFileChooser`

- Cuadro de diálogo para *abrir* un archivo: `showOpenDialog()`
- Botón seleccionado (Save/Cancel): `JFileChooser.APPROVE_OPTION`
- Archivo seleccionado (clase `File`): `getSelectedFile()`

AppletLeerArchivosV5

```
JFileChooser fChooser = new JFileChooser();
int result = fChooser.showOpenDialog(this);
if(result == JFileChooser.APPROVE_OPTION){
 File archivo = fChooser.getSelectedFile();
 try{
  FileReader fReader = new FileReader(archivo);
  BufferedReader bufReader = new BufferedReader(fReader);
  String linea = bufReader.readLine();
  while(linea != null){
   println(linea);
   linea = bufReader.readLine();
  }
  bufReader.close();

 }catch(FileNotFoundException e){
  println("EL archivo no se encontró");
 }catch(IOException e){
  println("EL archivo pudo leerse");
 }
}
```

Interfaz Gráfica para *Guardar* Archivos

Clase `JFileChooser`

Interfaz Gráfica para *Guardar* Archivos

Clase `JFileChooser`

Cuadro de diálogo para *guardar* un archivo: `showSaveDialog()`

Interfaz Gráfica para *Guardar* Archivos

Clase `JFileChooser`

Cuadro de diálogo para *guardar* un archivo: `showSaveDialog()`

Botón seleccionado (Save/Cancel):
`JFileChooser.APPROVE_OPTION`

Interfaz Gráfica para *Guardar* Archivos

Clase `JFileChooser`

- Cuadro de diálogo para *guardar* un archivo: `showSaveDialog()`
- Botón seleccionado (Save/Cancel): `JFileChooser.APPROVE_OPTION`
- Archivo seleccionado (clase `File`): `getSelectedFile()`

AppletEscribirArchivosV3

```
JFileChooser fChooser = new JFileChooser();
int result = fChooser.showSaveDialog(this);
if(result == JFileChooser.APPROVE_OPTION){
 File archivo = fChooser.getSelectedFile();
 try{
  FileWriter fWriter = new FileWriter(archivo);
  PrintWriter writer = new PrintWriter(fWriter);
  char c = 'A';
  for(int i= 0; i < 26; i++){
   writer.println((char)(c+i));
  }
  writer.close();
 }catch(IOException e){
  println("EL archivo pudo escribirse");
 }
}
```

Leyendo Archivos de Datos: Scanner

Clase Scanner

En vez de leer lineas como String...

Clase Scanner

En vez de leer lineas como String...

lee cada línea como un dato (int, double, boolean)

Clase Scanner

En vez de leer lineas como String...

lee cada línea como un dato (int, double, boolean)

Reemplaza a `BufferedReader` (requiere un `FileReader`)

```
FileReader fReader = new FileReader(archivo);
Scanner sc = new Scanner(fReader);
```

Clase Scanner

En vez de leer lineas como String...

lee cada línea como un dato (int, double, boolean)

Reemplaza a `BufferedReader` (requiere un `FileReader`)

```
FileReader fReader = new FileReader(archivo);
Scanner sc = new Scanner(fReader);
```

Métodos para saber si tiene al menos un dato más:

- `sc.hasNextInt()`: retorna verdadero si tiene al menos un dato tipo entero más

Clase Scanner

En vez de leer lineas como String...

lee cada línea como un dato (int, double, boolean)

Reemplaza a `BufferedReader` (requiere un `FileReader`)

```
FileReader fReader = new FileReader(archivo);
Scanner sc = new Scanner(fReader);
```

Métodos para saber si tiene al menos un dato más:

- `sc.hasNextInt()`: retorna verdadero si tiene al menos un dato tipo entero más

Métodos para leer el siguiente dato:

- `sc.nextInt()`: retorna el siguiente dato entero

AppletLeerArchivosV6

```
JFileChooser fChooser = new JFileChooser();
int result = fChooser.showOpenDialog(this);
if(result == JFileChooser.APPROVE_OPTION){
 File archivo = fChooser.getSelectedFile();
 try{
  FileReader fReader = new FileReader(archivo);
  Scanner sc = new Scanner(fReader);
  int suma = 0;
  while(sc.hasNextInt()){
   int entrada = sc.nextInt();
   suma = suma + entrada;
   println("entrada: "+entrada+ "\t suma:"+suma);
  }
  sc.close();
  fReader.close();

 }catch(FileNotFoundException e){
  println("EL archivo no se encontró");
 }catch(IOException e){
  println("EL archivo pudo leerse");
```

Clase Scanner con dobles

Se pueden leer dobles también

Clase Scanner con dobles

Se pueden leer dobles también

```
sc.hasNextDouble()
```

Clase Scanner con dobles

Se pueden leer dobles también

`sc.hasNextDouble()`

`sc.nextDouble()`:

Clase Scanner con dobles

Se pueden leer dobles también

`sc.hasNextDouble()`

`sc.nextDouble()`:

Cuidado especial con separador decimal: (punto o coma)

Clase Scanner con dobles

Se pueden leer dobles también

`sc.hasNextDouble()`

`sc.nextDouble()`:

Cuidado especial con separador decimal: (punto o coma)

- `sc.useLocale(Locale.ENGLISH)`: usa el estándar anglosajón (punto)
- `sc.useLocale(Locale.FRENCH)`: usa el estándar latino (coma)

AppletLeerArchivosV7

```
JFileChooser fChooser = new JFileChooser();
int result = fChooser.showOpenDialog(this);
if(result == JFileChooser.APPROVE_OPTION){
 File archivo = fChooser.getSelectedFile();
 try{
  FileReader fReader = new FileReader(archivo);
  Scanner sc = new Scanner(fReader);
  sc.useLocale(Locale.ENGLISH);
  double suma = 0;
  while(sc.hasNextDouble()){
   double entrada = sc.nextDouble();
   suma = suma + entrada;
   println("entrada: "+entrada+ "\t suma:"+suma);
  }
  sc.close();
  fReader.close();
 }catch(FileNotFoundException e){
  println("EL archivo no se encontró");
 }catch(IOException e){
  println("EL archivo pudo leerse");
```

AppletLeerArchivosV8

```
JFileChooser fChooser = new JFileChooser();
int result = fChooser.showOpenDialog(this);
if(result == JFileChooser.APPROVE_OPTION){
 File archivo = fChooser.getSelectedFile();
 try{
  FileReader fReader = new FileReader(archivo);
  Scanner sc = new Scanner(fReader);
  sc.useLocale(Locale.FRENCH);
  double suma = 0;
  while(sc.hasNextDouble()){
   double entrada = sc.nextDouble();
   suma = suma + entrada;
   println("entrada: "+entrada+ "\t suma:"+suma);
  }
  sc.close();
  fReader.close();
 }catch(FileNotFoundException e){
  println("EL archivo no se encontró");
 }catch(IOException e){
  println("EL archivo pudo leerse");
```