Práctica No. 6. Manejo de Excepciones

Java permite el manejo de excepciones, esto permite que cuando ocurra algún error en tiempo de ejecución, se pueda evitar que el programa finalice de manera incorrecta.

Una de las formas principales para este manejo es con la sentencia try-catch-finally que permite "capturar" y procesar los posibles errores que puedan aparecer.

Si bien hay gran cantidad de excepciones, todas derivan de la clase **Exception**, además de las excepciones ya manejadas, se pueden crear propias.

Se comenzará creando un nuevo proyecto llamado **Practica06** y la clase **Principal** en un paquete llamado **uam.pvoe.excepciones.principal**

Principal.java

```
public class Principal {

   public static void main(String[] args) {

      primeraExcepcion();

   }

}
```

Se crea un método llamado ***primeraExcepcion***

```
public static void primeraExcepcion(){
      String dato = "";
      int convertido = 0;
      Scanner teclado = new Scanner(System.in);

      System.out.println("Introduce un número, lo leeré como cadena y lo convertiré:");
      dato = teclado.nextLine();

      /*Cuando se quiere convertir una cadena que no es un número a entero,
      se produce un error
      */

      /*Dentro de la sentencia try se colocan las instruciones
      que podrían generar error
      */
      try{
         convertido = Integer.parseInt(dato);
      }
      catch(Exception e){
```

```
        /*Con catch se atrapa la excepción y se puede realizar alguna acción*/
        System.out.println("El dato leído no es un número!!");
    }finally{
      System.out.println("Se ejecuta haya excepciones o no, puede servir "
            + "para cerrar un archivo o una conexión a BD");
    }

    System.out.println("El dato convertido * 10 = "+convertido*10);

  }
```

Al introducir una cadena que es un número, el programa finaliza de manera adecuada, pero al introducir una cadena que no es un número, el programa finaliza con una excepción. Para evitar que el programa finalice de manera incorrecta se colocará un manejo de excepciones.

El bloque de ***try***, necesita forzosamente tener un bloque ya sea ***catch*** o ***finally***. Lo común es que se maneje solamente ***try/catch***. Instrucciones que se encuentren después del bloque ***catch***, también serán ejecutadas. Las instrucciones en un bloque ***finally*** también siempre serán ejecutadas.

Combinando Excepciones

Es posible combinar el uso de varias Excepciones, para esto se pueden colocar varios bloques try/catch o se puede colocar un solo bloque try con varios catch y que cada uno “capture” un tipo de excepción. Para ésto se creará un método llamado ***combinandoExcepciones***.

```
public static void combinandoExcepciones(){
    String datoLeido = "";
    String datoAux = null;
    int datoConvertido;

    Scanner teclado = new Scanner(System.in);
    datoLeido = teclado.nextLine();

    if(datoLeido.length()>5){
      datoAux = "hola";
    }
    try{
      int longitud = datoAux.length(); //Puede generar un error Null Pointer
      datoConvertido = Integer.parseInt(datoAux); //Puede generar un error de conversión
    }catch(NullPointerException e){
      System.out.println("Trabajando con un valor nulo!");
    }catch (NumberFormatException e){
      System.out.println("No se pudo realizar la conversión");
    }
  }
```

Lanzando y Atrapando Excepciones

Otras sentencias relacionadas con las excepciones son ***throw/catch***, las cuáles se utilizan cuando se llaman métodos que pueden generar una excepción. Son utilizados frecuentemente cuando se crean excepciones propias.

También se puede usar para crear un nuevo mensaje en una excepción ya existente. Cuando un método va a regresar una excepción, se indica con la sentencia ***throws*** seguido de la clase de la excepción (pueden ser varias) que se va a regresar.

Crear una método llamado ***convertir***

```
/*Método que lanza una excepción con un mensaje diferente*/
   public static int convertir(String dato) throws NumberFormatException{

      int convertido = 0;
          convertido = Integer.parseInt(dato);
       return convertido;

   }
```

Se creará un método llamado ***atrapandoExcepciones*** para invocar al que lanza una excepción.

```
public static void atrapandoExcepciones(){

      try{
         System.out.println("Convertido: " + convertir("Palabra"));
      }catch(NumberFormatException e){
         System.out.println("No puedo realizar la conversión");
      }
   }
```

Éste método se manda a llamar del método ***main***

```
public static void main(String[] args) {
      //primeraExcepcion();
      //combinandoExcepciones();
      atrapandoExcepciones();
   }
```

Creando una Excepción Propia

Las sentencias de ***throw/catch*** también permiten el manejo de excepciones propias. Lo primero es crear una excepción que herede de la clase ***Exception***. Para esto se creará un paquete nuevo llamado

uam.pvoe.excepciones.excepcionespropias.

Crear la clase **MiExcepcion.java** que herede de **Exception**

MiExcepcion.java

```
package uam.pvoe.excepciones.excepcionespropias;

public class MiExcepcion extends Exception {

   /*Constructor que regresa el mensaje que se quiere mostrar
   cuando ocurre la Excepción y se lo pasa al constructor de
   la clase Exception
   */
   public MiExcepcion(String mensajeError){
      super(mensajeError);
   }

}
```

En la clase **Principal** se colocarán dos métodos, uno para realizar la validación y lanzar una excepción y otro que lo mandará llamar.

```
public static void validaPropia(){

      int n= 5;

      try{
         minimo(n);
         System.out.println(n + " es mayor a 10");
      }catch(MiExcepcion e){
         System.out.println("No me sirve ese número");
      }
   }
```

```
/*Método que lanza una Excepción propia*/
   public static void minimo(int numero)throws MiExcepcion{

      if(numero <= 10){
         throw new MiExcepcion("Necesito un número mayor a 10");
      }
   }
```

Finalmente se manda a llamar el método ***validaPropia*** desde el método ***main***.

Mensajes de las Excepciones

Es posible desplegar varios de los mensajes generados por las excepciones, para esto se cambian las llamadas de ***System.out.println*** a ***System.err.println***

Los mensajes se obtienen del objeto creado a partir de la excepción manejada al momento de "atrapar" el error. También es posible imprimir un rastro de las líneas por las que se pasó antes de que se generara el error.

Se muestra este manejo en una de los manejos de excepciones del método ***main,*** invocando al método ***mensajeExcepciones.***

```
public static void main(String[] args) {


    //primeraExcepcion();
    //combinandoExcepciones();
    //validaPropia();
    mensajeExcepciones();
}
```

El siguiente método realiza un manejo de mensajes recibidos por una excepción.

```
/*Manejo de Excepciones*/
public static void mensajesExcepciones(){

    try{
        System.out.println("Convertido: " + convertir("Hola"));
    }catch(NumberFormatException e){

        System.out.println("No puede realizar la conversión");

        System.err.println(e.getMessage());
        System.err.println(e.getCause());

        StackTraceElement [] errores = e.getStackTrace();

        for(int i=0;i<errores.length;i++){
            System.err.println("ERROR: " + errores[i].toString());
        }
    }
}
```