Excepciones e hilos. Acceso a la red - Ejercicios

Índice

1. Captura de excepciones (*)

En el proyecto `java-excepciones` de las plantillas de la sesión tenemos una aplicación de Java en `Ej1.java` que toma un número como parámetro, y como salida muestra el logaritmo de dicho número. Sin embargo, en ningún momento comprueba si se ha proporcionado algún parámetro, ni si ese parámetro es un número. Se pide:

a) Compilar el programa y ejecutadlo de tres formas distintas:

- Sin parámetros

```
java Ej1
```

- Poniendo un parámetro no numérico

```
java Ej1 pepe
```

- Poniendo un parámetro numérico

```
java Ej1 30
```

Anotad las excepciones que se lanzan en cada caso (si se lanzan)

b) Modificar el código de `main` para que capture las excepciones producidas y muestre los errores correspondientes en cada caso:

- Para comprobar si no hay parámetros se capturará una excepción de tipo `ArrayIndexOutOfBoundsException` (para ver si el *array* de `String` que se pasa en el `main` tiene algún elemento).
- Para comprobar si el parámetro es numérico, se capturará una excepción de tipo `NumberFormatException`.

Así, tendremos en el `main` algo como:

```
try
{
        // Tomar parámetro y asignarlo a un double
} catch (ArrayIndexOutOfBoundsException e1) {
        // Codigo a realizar si no hay parametros
} catch (NumberFormatException e2) {
        // Codigo a realizar con parametro no numerico
}
```

Probad de nuevo el programa igual que en el caso anterior comprobando que las excepciones son capturadas y tratadas.

2. Lanzamiento de excepciones

El fichero `Ej2.java` es similar al anterior, aunque ahora no vamos a tratar las excepciones del `main`, sino las del método `logaritmo`: en la función que calcula el logaritmo se comprueba si el valor introducido es menor o igual que 0, ya que para estos

valores la función logaritmo no está definida. Se pide:

a) Buscar entre las excepciones de Java la más adecuada para lanzar en este caso, que indique que a un método se le ha pasado un argumento ilegal. (Pista: Buscar entre las clases derivadas de `Exception`. En este caso la más adecuada se encuentra entre las derivadas de `RuntimeException`).

b) Una vez elegida la excepción adecuada, añadir código (en el método `logaritmo`) para que en el caso de haber introducido un parámetro incorrecto se lance dicha excepción.

```
throw new ... // excepcion elegida
```

Probar el programa para comprobar el efecto que tiene el lanzamiento de la excepción.

c) Al no ser una excepción del tipo *checked* no hará falta que la capturemos ni que declaremos que puede ser lanzada. Vamos a crear nuestro propio tipo de excepción derivada de `Exception` (de tipo *checked*) para ser lanzada en caso de introducir un valor no válido como parámetro. La excepción se llamará `WrongParameterException` y tendrá la siguiente forma:

```
public class WrongParameterException extends Exception
{
        public WrongParameterException(String msg) {
                super(msg);
        }
}
```

Deberemos lanzarla en lugar de la escogida en el punto anterior.

```
throw new WrongParameterException(...);
```

Intentar compilar el programa y observar los errores que aparecen. ¿Por qué ocurre esto? Añadir los elementos necesarios al código para que compile y probarlo.

d) Por el momento controlamos que no se pase un número negativo como entrada. ¿Pero qué ocurre si la entrada no es un número válido? En ese caso se producirá una excepción al convertir el valor de entrada y esa excepción se propagará automáticamente al nivel superior. Ya que tenemos una excepción que indica cuando el parámetro de entrada de nuestra función es incorrecto, sería conveniente que siempre que esto ocurra se lance dicha excepción, independientemente de si ha sido causada por un número negativo o por algo que no es un número, pero siempre conservando la información sobre la causa que produjo el error. Utilizar *nested exceptions* para realizar esto.

Ayuda

Deberemos añadir un nuevo constructor a `WrongParameterException` en el que se proporcione la excepción que causó el error. En la función `logaritmo` capturaremos cualquier excepción que se produzca al convertir la cadena a número, y lanzaremos una excepción `WrongParameterException` que incluya la excepción causante.

3. Chat para el móvil

Vamos a ver como ejemplo una aplicación de chat para el móvil. En el directorio `ejemplos` de las plantillas de la sesión se encuentra una aplicación web con todos los servlets que necesitaremos para probar los ejemplos. Podremos desplegar esta aplicación en Tomcat para hacer pruebas con nuestro propio servidor.

Podemos encontrar la aplicación de chat implementada en el directorio `Chat`, que realiza las siguientes tareas:

- Lo primero que se mostrará será una pantalla de *login*, donde el usuario deberá introducir el *login* con el que participar en el chat. Deberemos enviar este *login* al servidor para iniciar la sesión. Para ello abriremos una conexión con la URL del *servlet* proporcionando los siguientes parámetros:

```
?accion=login&id=<nick_del_usuario>
```

  Si el *login* es correcto, el servidor nos devolverá un código de respuesta `200 OK`. Además deberemos leer la cabecera `URL-Reescrita`, donde nos habrá enviado la URL rescrita que deberemos utilizar de ahora en adelante para mantener la sesión.
- Una vez hemos entrado en el chat, utilizaremos la técnica de *polling* para obtener los mensajes escritos en el chat y mostrarlos en la pantalla. Utilizando la URL rescrita, conectaremos al *servlet* del chat proporcionando el siguiente parámetro:

```
?accion=lista
```

  Esto nos devolverá como respuesta una serie de mensajes, codificados mediante un objeto `DataOutputStream` de la siguiente forma:

```
<nick1> <mensaje1>
<nick2> <mensaje2>
...
<nickN> <mensajeN>
```

  De esta forma podremos utilizar un objeto `DataInputStream` para ir leyendo con el método `readUTF` las cadenas del *nick* y del texto de cada mensaje del chat:

```
String nick = dis.readUTF();
String texto = dis.readUTF();
```

- Para enviar mensajes al chat utilizaremos el bloque de contenido, conectándonos a la URL rescrita proporcionando el siguiente parámetro:

```
?accion=enviar
```

El mensaje se deberá codificar en binario, escribiendo la cadena del mensaje con el método `writeUTF` de un objeto `DataOutputStream`. Si obtenemos una respuesta `200 OK` el mensaje habrá sido enviado correctamente.