



APUNTE N°3: Introducción al manejo de excepciones y uso de packages con LDP Java

Prof. Dr. Samuel Sepúlveda
DCI-CEIS-UFRO

2016



CONTENIDOS

1. Manejo de excepciones
2. Uso de packages

1. INTRO MANEJO DE EXCEPCIONES CON JAVA

En Java, una excepción (Exception) es un cierto tipo de error que se ha producido en tiempo de ejecución (runtime).

Un “buen programa” debe ser capaz de gestionar correctamente los errores que se pueden producir.

1.1 Bloques try-catch

Las secciones a monitorear en busca de excepciones, están contenidas en un bloque: try-catch.

```
try { // código que queremos supervisar }  
    Si ocurre una excepción dentro de un try, ésta es lanzada y para capturarla se  
    usa el bloque:  
catch (Exception) { // código para gestionarla }
```

1.2 Consideraciones para un bloque try-catch

```
try {  
    // bloque a controlar en busca de excepciones  
}  
catch (tipoException-1 exc) {  
    // código que permite manipular la excepción }  
.....  
catch (tipoException-N exc) {  
    // código que permite manipular la excepción N }
```

De la figura anterior se tiene que:

- para una misma instrucción **try** se pueden capturar **n excepciones**, usando **n catch**.
- al cumplirse alguno de los catch, se ejecuta el código asociado a éste y el resto de ellos es omitido.
- sino es lanzada ninguna excepción dentro del bloque **try**, el programa ignora todos los **catch** y sigue su funcionamiento.

1.3 Tipos de excepciones

Existen diversos tipos de excepciones que pueden ser capturadas, entre ellas:

- IOException: clase general usada para capturar errores en operaciones de E/S. Se encuentra dentro del package **java.io**
- FileNotFoundException: clase que permite manipular el hecho de que se trató de abrir un archivo en una ruta no válida.
- AWTException: clase que indica un error de Abstract Window Toolkit ha sido generado. Se encuentra dentro del package **java.awt**

Se recomienda revisar más detalles sobre tipos de excepciones y su uso en:
<http://docs.oracle.com/javase/1.5.0/docs/api/java/lang/Exception.html>

2. Los package

Para nadie es un misterio que el elemento fundamental en una aplicación Java son las clases.

Sin embargo, existe un elemento de “mayor rango” llamado **package**, que es un contenedor de clases, similar al concepto de bibliotecas de C / C++.

Existen fundamentalmente por comodidad y diseño, pues nos permiten agrupar varios archivos fuente *.java en un sólo package.

2.1 Creación y uso de un package

Para su creación se incluye una línea de instrucción al comienzo del mismo con la ruta y nombre que le daremos a nuestro package. Veamos el siguiente ejemplo:

```
package net.programacion.ejemplos;
```

El código compilado (archivos *.class) se ubicarían en el directorio net/programacion/ejemplos.

Para poder utilizarlo desde otro programa se debe utilizar la sentencia **import**, que también se coloca al principio del programa (aunque después de package). Veamos un ejemplo:

```
import net.programacion.net.*;
```

Así, con el uso de los package podemos empezar a “independizar” nuestra clase pública de otras clases, ganando así en portabilidad.

2.2 Consideraciones

Hay que indicar que import es una palabra clave que indica las clases que se desean cargar, no los paquetes.

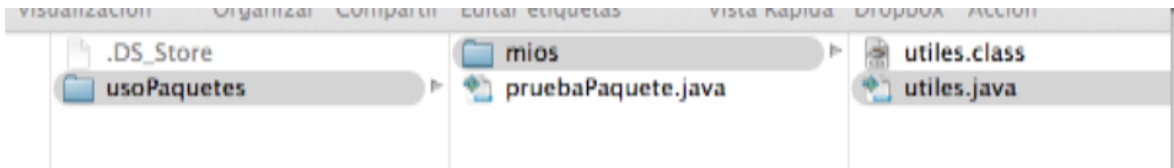
Sin embargo, al admitir el uso del comodín *, se puede utilizar para cargar paquetes enteros.

Por ejemplo, si se desea utilizar la clase **Calendar**, situada en el paquete **java.util**, se puede cargar de dos maneras:

- `import java.util.Calendar;`
- `import java.util.*;`

2.3 Ejemplo y visualización de la creación y uso de un package

i. **Package mios**, ubicado en la ruta: `/usoPaquetes/mios` y contiene la **clase Utiles.java**

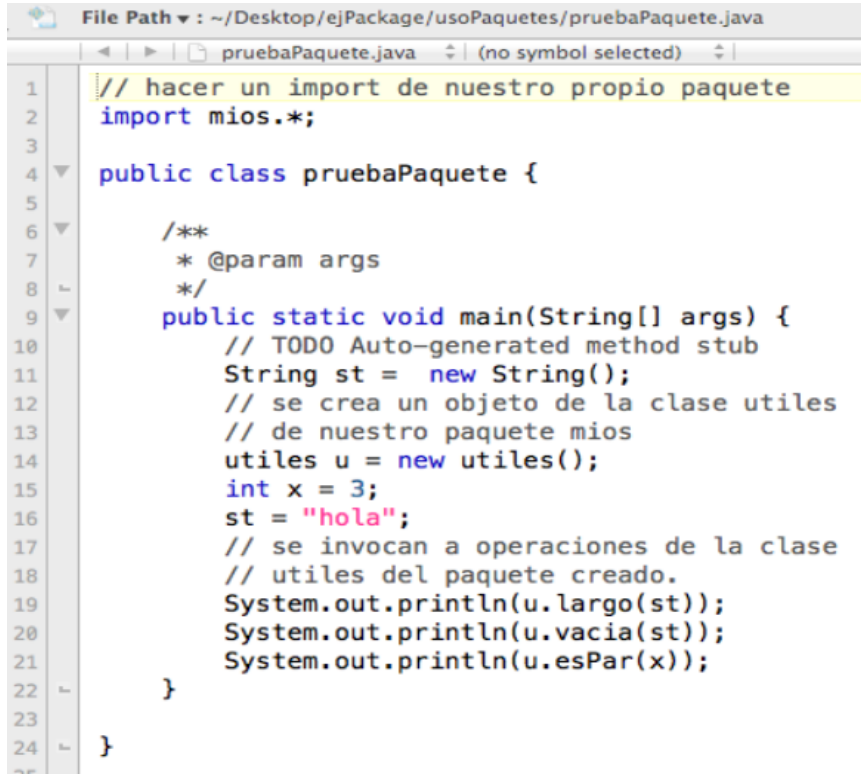


ii. Creación del **package mios**

```
File Path: ~/Desktop/ejPackage/usoPaquetes/mios/utiles.java
utiles.java (no symbol selected)

1  /******
2  * Ejemplo de la creación de un paquete personalizado
3  * clase con utilitarios que serán llamados después desde
4  * otros programas mediante un import.
5  * Este package debe ser guardado en un directorio con el
6  * mismo nombre del package.
7  */
8
9
10 // se define el nombre del paquete
11 package mios;
12
13 public class utiles {
14
15     public int largo(String s) {
16         return s.length();
17     }
18
19     public boolean vacia(String s) {
20         return (s.length() == 0);
21     }
22
23     public boolean esPar(int n) {
24         return (n%2 == 0);
25     }
26 }
27
```

iii. Inclusión de nuestro **package mios**, creado y uso de clases y métodos incluidos en él.



```
File Path : ~/Desktop/ejPackage/usoPaquetes/pruebaPaquete.java
pruebaPaquete.java (no symbol selected)

1 // hacer un import de nuestro propio paquete
2 import mios.*;
3
4 public class pruebaPaquete {
5
6     /**
7     * @param args
8     */
9     public static void main(String[] args) {
10         // TODO Auto-generated method stub
11         String st = new String();
12         // se crea un objeto de la clase utiles
13         // de nuestro paquete mios
14         utiles u = new utiles();
15         int x = 3;
16         st = "hola";
17         // se invocan a operaciones de la clase
18         // utiles del paquete creado.
19         System.out.println(u.largo(st));
20         System.out.println(u.vacia(st));
21         System.out.println(u.esPar(x));
22     }
23
24 }
```

Resumiendo

- Manejo de Excepciones en Java
 - Bloques try-catch
 - Ejemplo de uso de excepciones
- Los package
 - Creación y uso
 - Consideraciones