ACCESO-A-DATOS-TEMA-3.pdf

user_4383848

Acceso a datos

1º Desarrollo de Aplicaciones Multiplataforma

Estudios España

ACCESO A DATOS - TEMA 3
TRABAJO CON FICHEROS XML

1. ACCESO A DATOS CON DOM Y SAX:

Son herramientas que nos ofrecen la posibilidad de leer ficheros **XML**. Se dedican a verificar si son ficheros válidos desde el punto de vista sintáctico. → PARSERS

SAX	DOM
Basado en eventos.	Carga el fichero en memoria.
Va analizando nodo por nodo.	Búsqueda de *tags* hacia delante y hacia detrás.
En principio sin muchas restricciones de memoria, ya que no carga la totalidad del fichero.	Estructuras de árbol.
Rapidez en tiempo de ejecución.	Más lento en tiempo de ejecución.
Es solo de lectura.	Se pueden insertar o eliminar nodos.

2. CONVERSIÓN DE FICHEROS XML:

Parser tipo DOM:

```
DocumentBuilderFactory factory = DocumentBuilderFactory.newInstance();
   Instanciamos clase DOCUMENTBUILDERFACTORY
factory.setValidating(true);
   Att de Validación true / fichero que cargue esté bien validado
factory.setIgnoringElementContentWhitespace(true);
   true tambien a ignorar elementos con espacios en blanco
DocumentBuilder builder = factory.newDocumentBuilder();
   se crea el OBJETO DocumentBuilder
File file = new File("C:\\temp\\pruebas\\prueba.xml");
   Instanciamos nuevo fichero indicando la ruta.
Document doc = builder.parse(file);
   CARGAMOS FICHERO con el metodo builder.parse(file)
```

Parser tipo SAX:

Destacar que se instanciará, en el método parse, un **Handler** que será el responsable de ejecutar ciertas operaciones como iniciar elementos, operaciones con nodos, etc.
Es en la definición del Handler es donde debemos indicar las operaciones que realice nuestro analizador de código.

```
SAXParserFactory factory = SAXParserFactory.newInstance();

factory.setValidating(true);

SAXParser saxParser = factory.newSAXParser();

File file = new File("test.xml");

saxParser.parse(file, new DefaultHandler());
```

3. PROCESAMIENTO XML: XPath:

XPATH → Forma de búsqueda de información a través de un documento XML.

Características:

- Definición de estructuras: Define las distintas partes de un documento XML como un elemento, atributos, textos…
- Expresiones: Posee expresiones potentes.
- Funciones estándar: Librería muy completa de funcionalidades estándar de manipulación de datos.

Pasos al usar la librería XPATH:

1. Importamos los paquetes relacionados con dicha librería
2. Crearemos un objeto de la clase DocumentBuilder
3. Cargaremos un fichero o un flujo de datos.
4. Crearemos un objeto XPATH y una expresión.
5. Realizaremos una compilación de dicha expresión con el método Xpath.compile() y obtendremos una lista de los nodos evaluando la expresión previamente compilada usando Xpath.evaluate()
6. Realizaremos una iteración por lo general de la lista de nodos.
7. Examinaremos los atributos
8. Examinaremos los sub elementos.
9. La instanciación del objeto Xpath la realizaremos de la siguiente forma:

XPath xPath = XPathFactory.newInstance().newXPath(); → **INSTANCIACIÓN OBJETO XPATH**

4. EXCEPCIONES:

Excepción: Evento que ocurre durante la ejecución de un programa, y que interrumpe el flujo del mismo por algún motivo.

- **Excepciones con chequeo** (checked exceptions): Son notificadas por el compilador en tiempo de compilación, no pueden ser ignoradas, y fuerzan al programador a manejarlas.

- **Excepciones sin chequeo** (unchecked exceptions): Se originan en tiempo de ejecución, son llamadas también RuntimeExceptions.

- **Errores:** No son del todo excepciones, escapan del control del usuario o del propio desarrollador. Los errores generalmente se ignoran en el código porque rara vez se puede hacer algo al respecto.

Excepciones asociadas a clases XML I:

- **getMessage():** Mensaje detallado sobre la excepción que se acaba de lanzar. El mensaje es instanciado en el constructor de la clase Throwable.
- **getCause():** Devuelve la causa representada en un objeto Throwable.
- **toString():** Devuelve el nombre de la clase y se le concatena el resultado de getMessage().
- **printStackTrace():** Imprime el resultado del método toString() junto con el error de sistema que devuelve la pila.
- **getStackTrace():** Devuelve un array con cada uno de los elementos de la pila. El elemento 0 del array representa el elemento más alto de la pila.
- **fillInstackTrace():** rellena la pila del objeto Throwable con la pila actual. Le añade cualquier información previa en el seguimiento de la misma.

Excepciones asociadas a clases XML II:

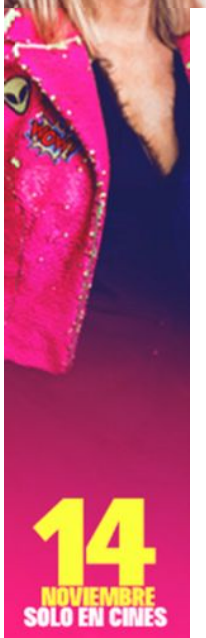
Añadir la excepción a la definición del método: con esta opción lo que estaremos haciendo será lanzar la excepción a un nivel superior.
public static void main(String[] args) throws ParserConfigurationException {

Rodear con sentencia try/catch: Rodearemos el código con la estructura básica try/catch.

Excepciones asociadas a clases XML III:

```
try {
//Distintas operaciones parse

} catch (ParserConfigurationException e) {
  e.printStackTrace();
} catch (IOException e) {
  e.printStackTrace();
} catch (XPathExpressionException e) {
  e.printStackTrace();
} catch (SAXException e) {
  e.printStackTrace();
}
```



5. PRUEBAS Y DOCUMENTACIÓN - JUnit:

Valida si los resultados de un código están en el estado que se espera, o ejecuta la secuencia esperada de operaciones o eventos. Ayudan al programador a verificar que un fragmento de código es correcto.

JUnit es un framework que usa anotaciones para identificar diferentes tests.
JUnit método → Clase Test class.
Para definir que un método forma parte de un test se tendrá que añadir la anotación **@Test** sobre la cabecera del método.

Existen algunas anotaciones y algunos métodos que debemos tener en cuenta para la escritura de test unitarios, como por ejemplo:

• **@Before:** Al inicio de la clase se definirá un método. Su utilidad será instanciar la mayor parte de las variables que vamos a necesitar para los test.
Siempre que ejecutemos un test unitario, antes, se ejecutará el código de este método con anotación @Before.

• **@After:** Con esta anotación, se definirá un método cuyo código se ejecutará, siempre después de finalizar cualquier test dentro de nuestra clase.