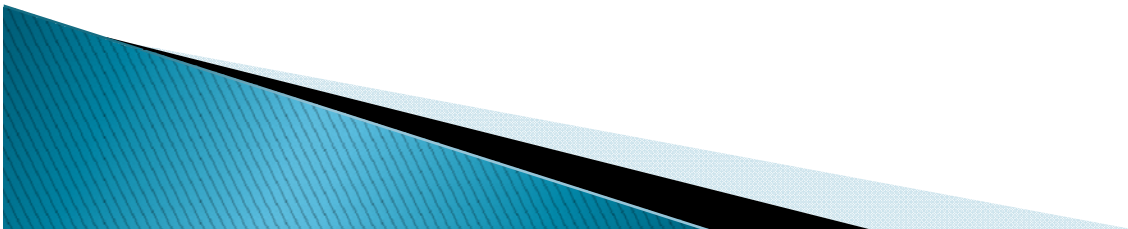


Índice

1. Introducción
2. Pruebas unitarias: JUnit
3. ANT
4. JMeter
5. Pruebas unitarias en aislamiento: JMock
6. Pruebas de base de datos: DBUnit



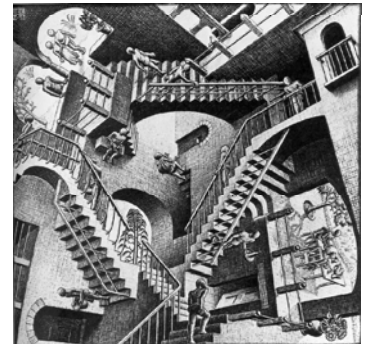
5. DBUnit

- ▶ Hoy en día, la mayoría de aplicaciones tienen persistencia en base de datos.
- ▶ Dos técnicas posibles para evaluar las bases de datos:
 - Uso de Jmocks
 - Uso de bases de datos reales



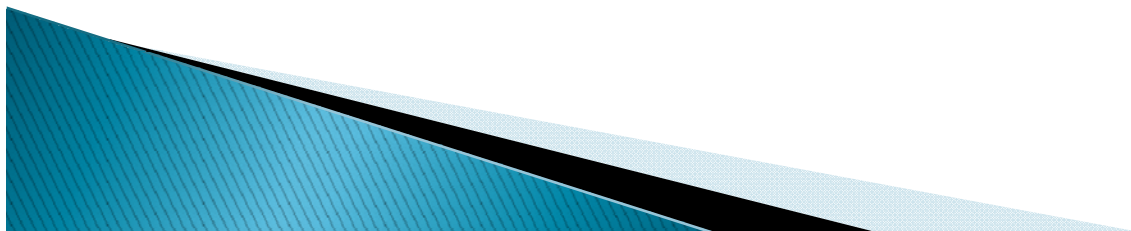
5. DBUnit

- ▶ El uso de JMock tiene algunas ventajas
 - Tiempo de ejecución de las pruebas es inferior. No hay conexión real con la BDA.
 - No requiere crear y configurar la BDA.
 - Permite probar situaciones excepcionales.
 - Se prueba el funcionamiento para cualquier driver de acceso a BDA.
- ▶ Y algunos inconvenientes:
 - La creación de los mocks lleva tiempo.
 - Hay que sincronizar BDA y pruebas si se modifica la BDA.
 - Realmente no se está probando la BDA.



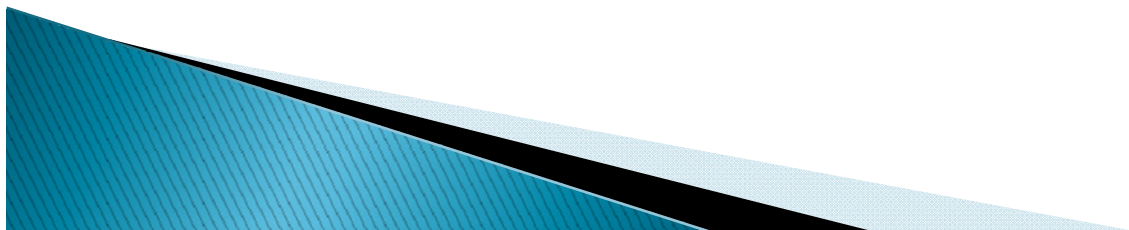
5. DBUnit

- ▶ Existen 3 tipos de BDA reales:
 - BDA de producción: BDA sobre la que trabaja la aplicación. Mejor no tocar para las pruebas.
 - BDA de pruebas: idéntica a la de producción y sobre la que se pueden hacer las pruebas.
 - BDA de datos de integración: sirve para verificar que los cambios sobre la BDA de pruebas se han aplicado también a la de producción.



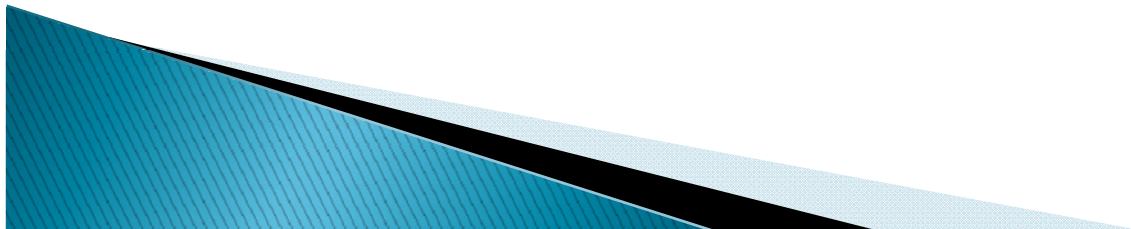
5. DBUnit

- ▶ Ventajas de trabajar con BDAs en las pruebas:
 - Se puede medir la temporización de acceso.
 - Podemos evaluar aspectos como los drivers de acceso a la BDA.
- ▶ Inconvenientes de trabajar con BDAs en las pruebas:
 - Es necesario crear, configurar y mantener la BDA.
 - El tiempo de ejecución de los casos de prueba se puede incrementar.
 - Hay situaciones excepcionales que es complicado de simular con la BDA real.



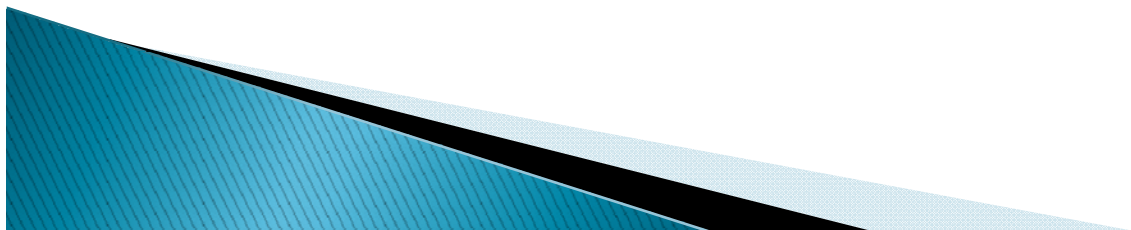
5. DBUnit

- ▶ Los pasos habituales en el proceso de desarrollo de aplicaciones grandes son:
 - Empezar simulando la BDA con mocks.
 - En paralelo, ir probando las clases con la BDA de pruebas.
 - Hacer pruebas de integración



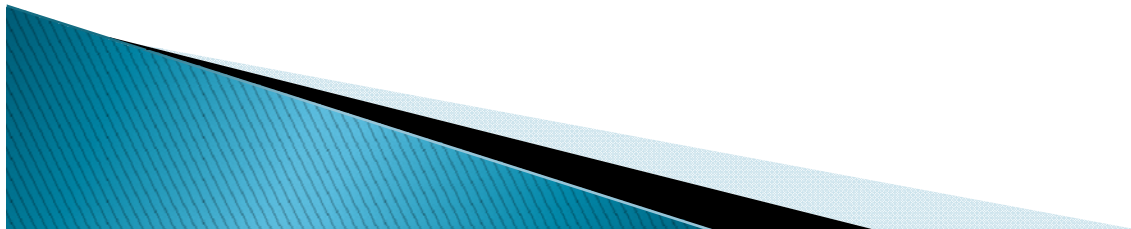
5. DBUnit

- ▶ JUnit ha desarrollado la herramienta DBUnit para probar el acceso a la BDA. La técnica se resume en:
 - Inicializar el contenido de la BDA de acuerdo al caso de prueba.
 - Ejecutar el método a probar. Normalmente implica alguna sentencia SQL.
 - Comparar el contenido de la BDA con el contenido esperado de acuerdo al caso de prueba.



5. DBUnit

- ▶ Ventajas del uso de DBUnit:
 - Proporciona un mecanismo para almacenar y representar conjunto de datos (dataset) procedentes de la BDA.
 - Es capaz de hacer consultas a la BDA y plasmar los resultados como un dataset que puede ser utilizado en los casos de prueba.
 - Proporciona mecanismos para la comparación de datasets. Muy utilizado para comparar la BDA con lo esperado.
- ▶ DBUnit se basa en un sistema de importación/exportación de los datos de la base de datos a un fichero XML (dataset)



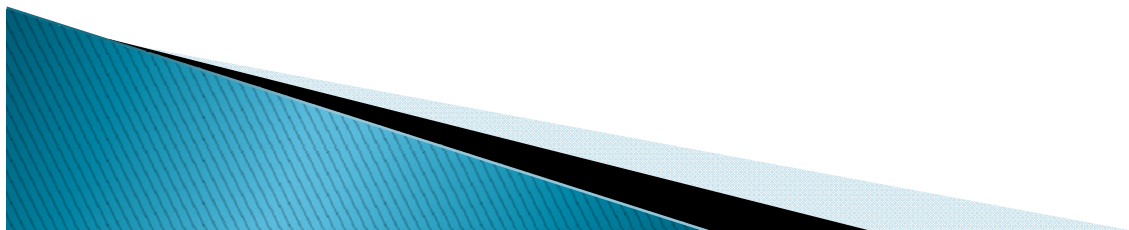
5.1 Creación de las pruebas

- ▶ Los componentes principales son:
 - `IDatabaseConnection`: Interfaz que representa una conexión DBUnit a la base de datos.
 - `IDataSet`: Interfaz que representa una colección de tablas (Manipula tablas y datos).
 - `DatabaseOperation`: Clase abstracta que representa la operación que se va a realizar sobre la base de datos antes o después de un test.



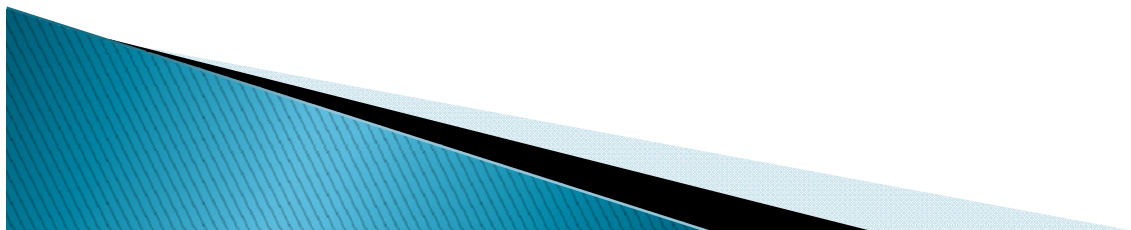
5.1 Creación de las pruebas

- ▶ El procedimiento básico para realizar un test es el siguiente:
 1. Creamos el DataSet de la prueba, preferiblemente en XML
 2. Configurar la conexión a la base de datos en `@BeforeClass`
 3. Inicializar la base de datos de pruebas con los datos de pruebas con `FlatXmlDataSet` en `@BeforeClass`
 4. Poner la base de datos a un estado definido en `@BeforeClass` usando `DatabaseOperation`
 5. Ejecutar el código a probar en `@Test` y verificar los resultados obtenidos
 6. Cerciorarse de verificar que no hay efectos no deseados y volver al estado definido en la base de datos en `@After` usando `DatabaseOperation`
 7. Eliminar las conexiones con la base de datos en `@AfterClass`



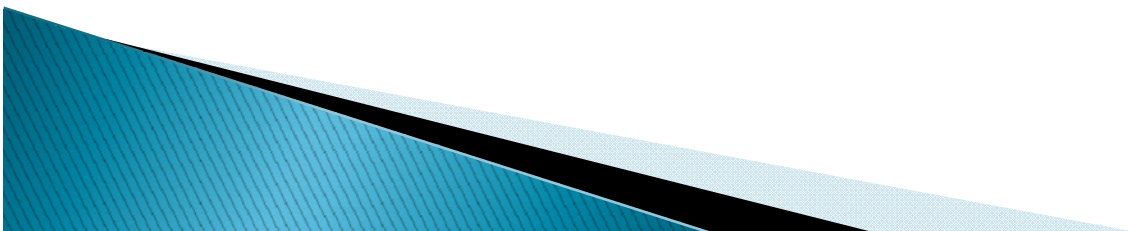
5.1 Creación de las pruebas

- ▶ Para conectarnos a la BDA de prueba:
 - Usar el driver “com.mysql.jdbc.Driver”
 - Especificar la URL de conexión con la BDA “jdbc:mysql://localhost:3306/nombre_BDA”.
 - Especificar nombre de usuario y contraseña para conectarse a la BDA (debería ir encriptado desde un fichero).



5.1 Creación de las pruebas

- ▶ Operaciones de DatabaseOperation
 - DatabaseOperation.UPDATE: actualiza la base de datos en base al dataset.
 - DatabaseOperation.INSERT: introduce el dataset en la base de datos.
 - DatabaseOperation.DELETE: solo borra el contenido del dataset en la base de datos.
 - DatabaseOperation.DELETE_ALL: borra todo el contenido de la base de datos.
 - DatabaseOperation.CLEAN_INSERT: hace la acción DELETE_ALL seguido de INSERT.



5.2 DataSet

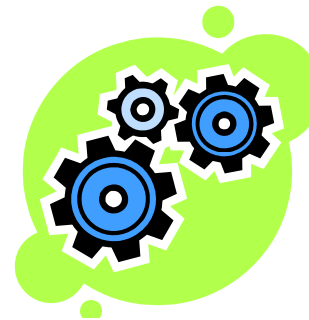
- ▶ Los DataSet son archivos xml, para cargar los datos de prueba en la base de datos, donde el nombre de los elementos se corresponde con el nombre de la tabla, y los atributos se corresponden con las columnas.

▶ Ej:

```
01  
02 <?xml version='1.0' encoding='UTF-8'?>  
03 <dataset>  
04   <USUARIO usuario_id='1'  
05     fecha_nacimiento='2001-01-01'  
06     nombre='Pepe'  
07     apellidos='Perez' />  
08   <USUARIO usuario_id='2'  
09     fecha_nacimiento='1991-11-02'  
10     nombre='Juan'  
11     apellidos='Garccía' />  
12 </dataset>  
13
```

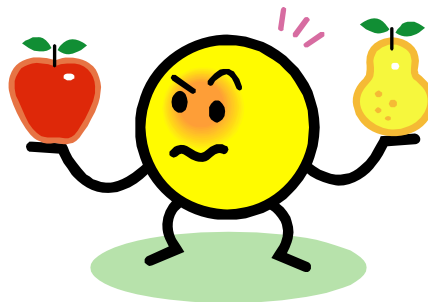
5.2 DataSet

- ▶ Los dataset se pueden crear manualmente si los datos de prueba no son muchos.
- ▶ Para gran cantidad de datos de prueba, se puede crear una función JAVA que exporte la base de datos al dataset (ver clase ExtractDataSet)



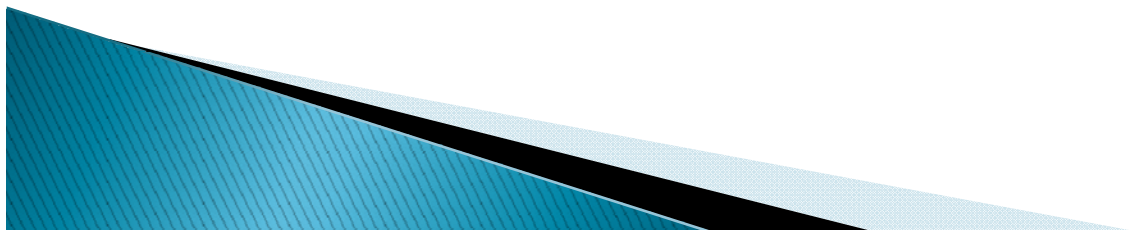
5.2 DataSet

- ▶ Dependiendo de los casos de prueba, nos puede interesar comparar el dataset de la BDA tras las pruebas con el dataset esperado.
- ▶ Para ello, se debe usar la clase **DbUnitAssert**.
- ▶ En concreto, el método:
 - `assertEquals(IDataSet expectedDataSet, IDataSet actualDataSet)`



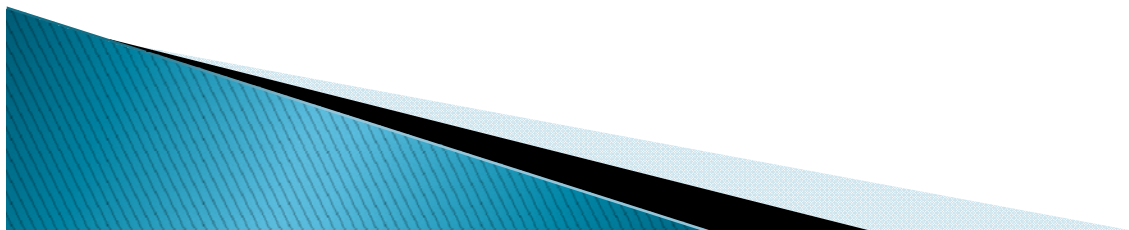
5.3 Definición de métodos de prueba

- ▶ El procedimiento es el siguiente:
 - Se inicializa la BDA de forma que contenga información conocida.
 - Se ejecuta el método a probar conforme a los datos definidos para el caso de prueba.
 - Se obtiene la información contenida en la BDA y la información que se espera contenga la BDA tras la ejecución del método a probar.
 - Se comparan ambas informaciones y si existe alguna diferencia se da la prueba como fallida.



5.3 Definición de métodos de prueba

- ▶ En un proyecto real, existe una batería de pruebas para cada clase.
- ▶ Cada clase JAVA debe tener un fichero de pruebas independiente.
- ▶ Se recomienda que exista una dataset específico para cada una de las clases JAVA a probar.
- ▶ Incluso pueden haber varios datasets para una misma clase, cargando cada uno de ellos en base a las necesidades.





















5.4 Recomendaciones

- ▶ El acceso a la BDA debe estar diferenciado del resto del código. Se recomienda tener una clase que haga la conexión y acceso.
- ▶ Asegurarse que el diseño de la BDA es el correcto.
- ▶ Utilizar una BDA local para cada desarrollador.
- ▶ Nunca crear casos de prueba que dependan del estado anterior que haya dejado otro caso de prueba.
- ▶ Los datasets deberían ser lo más pequeños posibles.
- ▶ Evitar el uso de procedimientos de almacenado.



5.5 Librerías a añadir

- +...  dbunit-2.4.9.jar
- +...  h2-1.3.173.jar
- +...  mysql-connector-java-5.1.20-bin.jar
- +...  slf4j-api-1.7.5-sources.jar
- +...  slf4j-api-1.7.5.jar
- +...  slf4j-ext-1.7.5-sources.jar
- +...  slf4j-ext-1.7.5.jar
- +...  slf4j-jdk14-1.7.5-sources.jar
- +...  slf4j-jdk14-1.7.5.jar
- +...  slf4j-migrator-1.7.5.jar
- +...  slf4j-nop-1.7.5-sources.jar
- +...  jcl-over-slf4j-1.7.5-sources.jar
- +...  jcl-over-slf4j-1.7.5.jar
- +...  jul-to-slf4j-1.7.5-sources.jar
- +...  jul-to-slf4j-1.7.5.jar
- +...  log4j-over-slf4j-1.7.5-sources.jar
- +...  log4j-over-slf4j-1.7.5.jar
- +...  JDK 1.7 (Default)