

# Informe PTI

Lab 3

Autores  
Marc Cabezas  
Antonio Arellano

<b>1. Familiarización con el entorno</b>	<b>3</b>
<b>2. Caso práctico</b>	<b>4</b>
2.1.CarRental.java	4
<b>3. Conclusiones</b>	<b>7</b>
3.1 Conclusiones de la familiarización con el entorno	7
3.2 Conclusiones del caso práctico	7
3.3 Conclusiones finales	7

# 1. Familiarización con el entorno

Se ha configurado la máquina, instalando Java y GIT para poder descargar correctamente los archivos y poder ejecutar los ficheros .java.

Se han ejecutado estos comandos:

```
sudo apt-get install git
```

```
cd $HOME
```

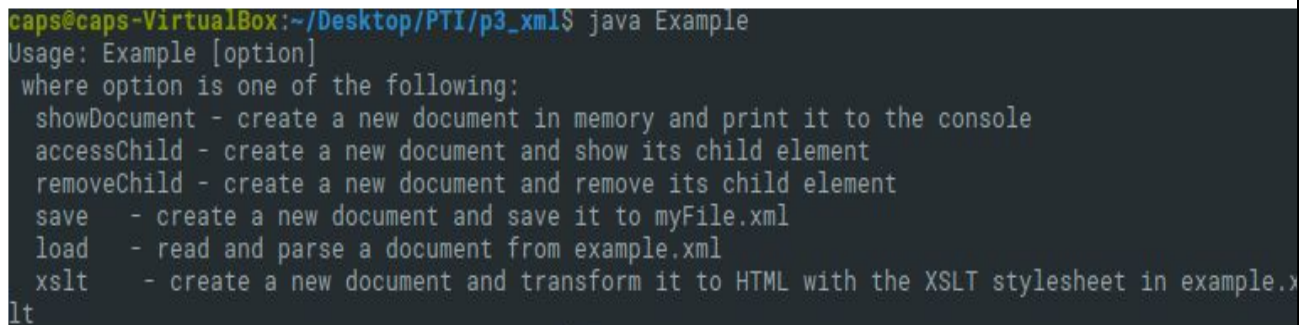
```
git clone https://gitlab.fib.upc.edu/pti/pti.git
```

```
sudo apt-get install default-jdk
```

```
export CLASSPATH=./xalan.jar:./xercesImpl.jar:./jdom.jar:
```

Una vez estaba todo instalado se han realizado pruebas para comprobar que todo se había instalado correctamente.

Se han realizado más pruebas con varios ejemplos:



```
caps@caps-VirtualBox:~/Desktop/PTI/p3_xml$ java Example
Usage: Example [option]
where option is one of the following:
  showDocument - create a new document in memory and print it to the console
  accessChild  - create a new document and show its child element
  removeChild  - create a new document and remove its child element
  save         - create a new document and save it to myFile.xml
  load         - read and parse a document from example.xml
  xslt         - create a new document and transform it to HTML with the XSLT stylesheet in example.x
lt
```

Figura 1. Salidas realizadas con el fichero de pruebas Example.java

Se han realizado todas las pruebas en el fichero de ejemplo para comprobar el buen funcionamiento de todos los métodos y de su lógica.

## 2. Caso práctico

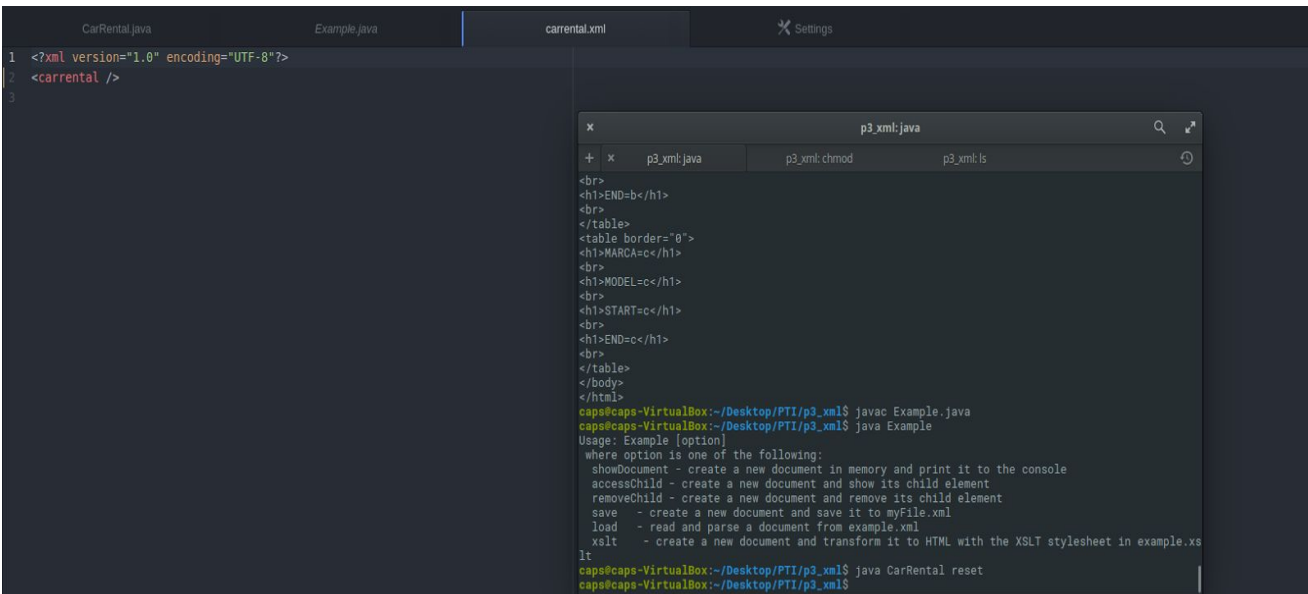
En esta sección se explicarán las soluciones adoptadas para los problemas propuestos, luego éstas soluciones nos ayudarán a elaborar un conclusión.

### 2.1.CarRental.java

Se nos pedía crear un programa con el cual se pudiera hacer diferentes acciones respecto un fichero xml.

Las acciones son las siguientes:

**-Reset.** El cual nos permite vaciar el fichero xml y dejarlo listo para que pueda ser usado de nuevo.



```
CarRental.java Example.java carrental.xml Settings
1 <?xml version="1.0" encoding="UTF-8"?>
2 <carrental />
3

x p3_xml.java
+ x p3_xml.java p3_xml:chmod p3_xml:ls
<br>
<h1>END=bc</h1>
<br>
</table>
<table border="0">
<h1>MARCA=c</h1>
<br>
<h1>MODEL=c</h1>
<br>
<h1>START=c</h1>
<br>
<h1>END=c</h1>
<br>
</table>
</body>
</html>
caps@caps-VirtualBox:~/Desktop/PTI/p3_xml$ javac Example.java
caps@caps-VirtualBox:~/Desktop/PTI/p3_xml$ java Example
Usage: Example [option]
where option is one of the following:
showDocument - create a new document in memory and print it to the console
accessChild - create a new document and show its child element
removeChild - create a new document and remove its child element
save - create a new document and save it to myFile.xml
load - read and parse a document from example.xml
xslt - create a new document and transform it to HTML with the XSLT stylesheet in example.xs
lt
caps@caps-VirtualBox:~/Desktop/PTI/p3_xml$ java CarRental reset
caps@caps-VirtualBox:~/Desktop/PTI/p3_xml$
```

Figura 2. Fichero XML después de efectuar la acción Reset.

**-New.** Con esta funcionalidad podremos añadir los alquileres que se vayan efectuando. Tendremos varios pasos que seguir:

- Añadir la marca del vehículo.
- Añadir el modelo del vehículo.
- Añadir la fecha de Inicio.
- Añadir la fecha de Fin.

```
1 <?xml version="1.0" encoding="UTF-8"?>
2 <carrental>
3   <rental id="902724">
4     <marca>a</marca>
5     <model>a</model>
6     <start>a</start>
7     <end>a</end>
8   </rental>
9   <rental id="703106">
10    <marca>b</marca>
11    <model>b</model>
12    <start>b</start>
13    <end>b</end>
14  </rental>
15  <rental id="982545">
16    <marca>c</marca>
17    <model>c</model>
18    <start>c</start>
19    <end>c</end>
20  </rental>
21 </carrental>
22
```

```
</html>
caps@caps-VirtualBox:~/Desktop/PTI/p3_xml$ java Example.java
caps@caps-VirtualBox:~/Desktop/PTI/p3_xml$ java Example
Usage: Example [option]
where option is one of the following:
  showDocument - create a new document in memory and print it to the console
  accessChild - create a new document and show its child element
  removeChild - create a new document and remove its child element
  save - create a new document and save it to myFile.xml
  load - read and parse a document from example.xml
  xslt - create a new document and transform it to HTML with the XSLT stylesheet in example.xs
It
caps@caps-VirtualBox:~/Desktop/PTI/p3_xml$ java CarRental reset
caps@caps-VirtualBox:~/Desktop/PTI/p3_xml$ java CarRental new
Introdueixi la marca del vehicle:a
Introdueixi el model del vehicle:a
Introdueixi la data d'inici del lloguer (format dd/mm/yyyy):a
Introdueixi la data de fi del lloguer (format dd/mm/yyyy):a
caps@caps-VirtualBox:~/Desktop/PTI/p3_xml$ java CarRental new
Introdueixi la marca del vehicle:b
Introdueixi el model del vehicle:b
Introdueixi la data d'inici del lloguer (format dd/mm/yyyy):b
Introdueixi la data de fi del lloguer (format dd/mm/yyyy):b
caps@caps-VirtualBox:~/Desktop/PTI/p3_xml$ java CarRental new
Introdueixi la marca del vehicle:c
Introdueixi el model del vehicle:c
Introdueixi la data d'inici del lloguer (format dd/mm/yyyy):c
Introdueixi la data de fi del lloguer (format dd/mm/yyyy):c
```

Figura 3. Fichero XML después de efectuar la acción New.

-**List.** Con esta funcionalidad podremos sacar por consola los alquileres que se hayan efectuado.

```
1 <?xml version="1.0" encoding="UTF-8"?>
2 <carrental>
3   <rental id="902724">
4     <marca>a</marca>
5     <model>a</model>
6     <start>a</start>
7     <end>a</end>
8   </rental>
9   <rental id="703106">
10    <marca>b</marca>
11    <model>b</model>
12    <start>b</start>
13    <end>b</end>
14  </rental>
15  <rental id="982545">
16    <marca>c</marca>
17    <model>c</model>
18    <start>c</start>
19    <end>c</end>
20  </rental>
21 </carrental>
22
```

```
caps@caps-VirtualBox:~/Desktop/PTI/p3_xml$ java CarRental new
Introdueixi la marca del vehicle:c
Introdueixi el model del vehicle:c
Introdueixi la data d'inici del lloguer (format dd/mm/yyyy):c
Introdueixi la data de fi del lloguer (format dd/mm/yyyy):c
caps@caps-VirtualBox:~/Desktop/PTI/p3_xml$ java CarRental list
<?xml version="1.0" encoding="UTF-8"?>
<carrental>
  <rental id="902724">
    <marca>a</marca>
    <model>a</model>
    <start>a</start>
    <end>a</end>
  </rental>
  <rental id="703106">
    <marca>b</marca>
    <model>b</model>
    <start>b</start>
    <end>b</end>
  </rental>
  <rental id="982545">
    <marca>c</marca>
    <model>c</model>
    <start>c</start>
    <end>c</end>
  </rental>
</carrental>
caps@caps-VirtualBox:~/Desktop/PTI/p3_xml$
```

Figura 4. Salida de los alquileres por consola.

-**Xslt**. Con esta funcionalidad podremos hacer uso de la tecnología XSLT y transformar nuestro fichero XML en un formato HTML.

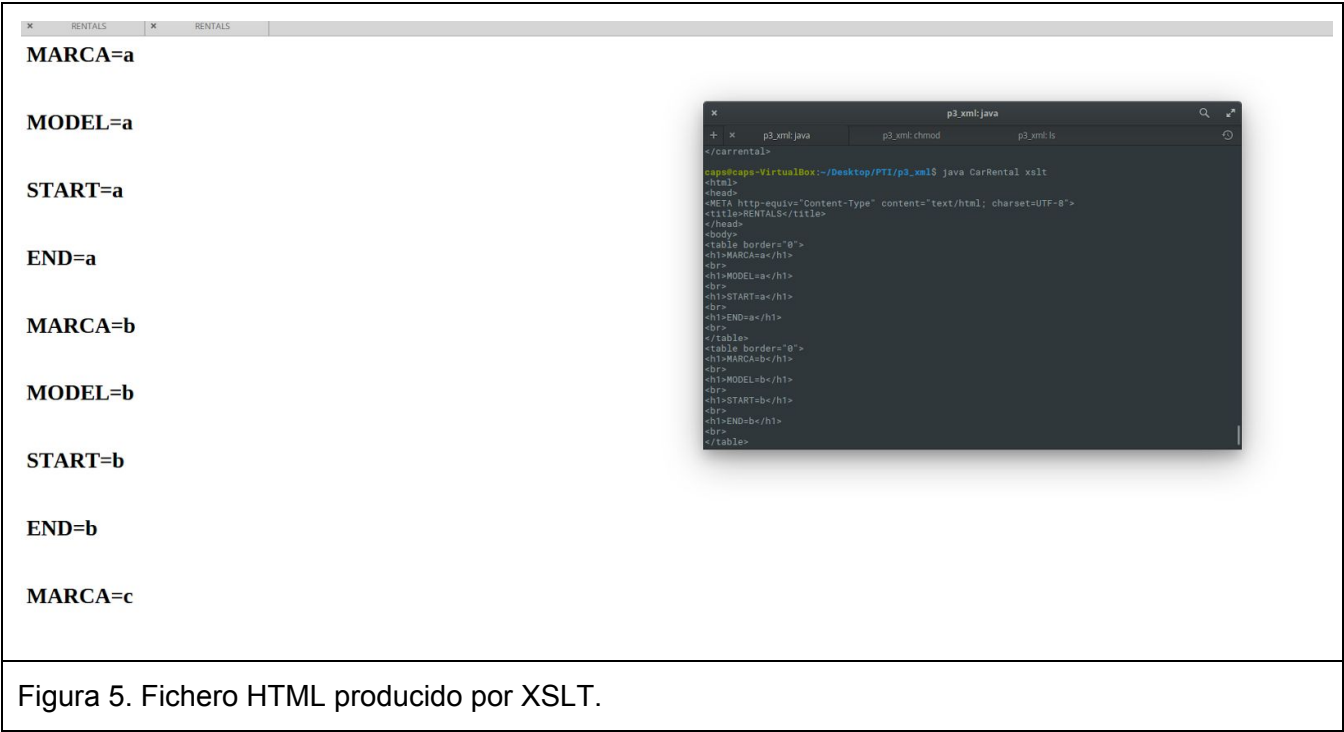


Figura 5. Fichero HTML producido por XSLT.

## 3. Conclusiones

Las conclusiones se vuelven a estructurar de la siguiente manera, primero comentaremos las conclusiones que hemos sacado del punto 1 (Familiarización con el entorno), luego las del punto 2 (Caso práctico) y finalmente a partir de las conclusiones anteriores, intentaremos sacar una conclusión definitiva de las tecnologías con las que hemos trabajado, así como puntos a favor y en contra o posibles escenarios en donde éstas tecnologías serían de gran utilidad. Se comparará con JSON con el fin de que las conclusiones sean constructivas.

### 3.1 Conclusiones de la familiarización con el entorno

La instalación de los paquetes necesarios no ha dado problemas. En la prueba de los ejemplos no ha habido ningún problema. Un punto positivo es el entorno de desarrollo, al ser una aplicación Java, la mayoría de IDEs permiten una rápida implementación de aquello que se necesite.

### 3.2 Conclusiones del caso práctico

No ha sido difícil entender cómo poder generar, cargar en memoria y modificar, una vez cargados, archivos xml con Java, el ejemplo proporcionado era bastante claro y tan solo se han tenido que adaptar ciertas funciones, o pensar la lógica de otras no muy complejas. Tratar elementos xml en Java no es complejo en cuanto se entiende la estructura y como Java trata estos elementos como si fueran objetos, una vez se tiene claro eso, es tan solo jugar con objetos. Si se quiere comparar con JSON y cómo lo trata Java, se podría decir que trabajar con JSON es mucho más amigable al programador, pero xml nos ofrece más potencia y libertad. A parte de la clase Java también se tuvo que adaptar la hoja de estilos que se nos proporcionaba con el ejemplo, con el fin de utilizarla para convertir el xml a html, con JSON este tipo de cosas se complican.

### 3.3 Conclusiones finales

Basándonos en las conclusiones anteriores y el informe de la semana pasada, podemos concluir que trabajar con xml nos ofrece más libertad a bajo nivel, a diferencia de con JSON que queda todo más reglado, para la mayoría de problemas que nos pudiésemos encontrar se pueden solucionar con JSON, pero si lo que se necesita es algo concreto y vemos que trabajar con xml nos facilita las cosas a posteriori, apostaremos por su uso, en temas de escalabilidad están bastante a la par, por lo que, si se quiere un desarrollo rápido y amigable elegiremos JSON por comodidad, pero si se quiere para un proyecto más grande y que además requiera xml, no huiremos de su uso ni buscaremos alternativas.