



Abstract 3

1. Introduction .....4

In

# Abstract





3/21/2002

For

## 2. Overview of CTF

In this section you will be given a quick overview of CTF: what is the main idea behind it? Where are the tests located? What are the test patterns? How to run the tests?

Note: currently CTF is written to test Castor XML; a future could contain the entire tests of Castor.

### 2.1 Creating a JUnit test case



3/21/2002



## Marshalling Framework testing

2 main patterns are used to test the marshalling framework.

### Marshal/Unmarshal process

- Instantiate an object model with random data or hardcoded data if any.
- Marshal it
- Compare the marshalled object to a 'gold' file if any
- Unmarshal the document obtained from the marshalling process
- Compare the unmarshal object to the original object model

### Unmarshal/Marshal process

- Unmarshal a provided XML document (input document)
- Compare the content of the unmarshalled object with hardcoded data
- Marshal the object model obtained
- Compare the obtained document with the input document

## Mapping

Testing the mapping functionalities of Castor is simply testing the Marshalling







For instance to run the tests for the Source Generator:

```
CTFRun -verbose pathofMasterTestSuite/xml/SourceGenerator
```

This command will execute in GUI mode (default mode) all the test cases written for the Source Generator and will print detailed messages about the execution of the tests.



3/21/2002

3



## 3.2 The directory



## Header

The header of the xml file contains all the generic information about a test:

For instance,

```
<?xml version='1.0'?>
<TestDescriptor>
  <Name>Bug in the support of foo</Name>
  <Comment>This test case illustrates the bad handling of the foo element in the
Marshalling Framework</Comment>
  <Category>basic capability</Category>
  <BugFix>
```



3/21/2002



3/21/2002

## UnitTestCase

This is the core of the test, this element provides the input file and output file used for a specific test and the name of the class used to instantiate the Object Model.

Once





## Example file

Here is a complete TestDescriptor.xml

```
<?xml version='1.0'?>
<TestDescriptor>
    <Name>Bug in the support of foo</Name>
    <Comment>This test case illustrates the bad handling of the foo element
in the Marshalling Framework</Comment>

<Date_Report>2001-03-11T12:00:00</Date_Report>
```

2.6



3/21/2002

## 3.4



```
*/  
  
public String dumpFields();  
  
/**  
 * The instance of the object will randomize the content  
 * of its field. This is used to create an instance  
 * of the object model without having to  
 * unmarshal anything.  
 */  
  
public void randomizeFields()  
    throws InstantiationException, IllegalAccessException ;
```

his mplemtensedyou() Tj -06 -150 TD0.0820 Tc haer tosetl 18atStrbutes 'dum'l ad 'rando'l to





```
public boolean equals(java.lang.Object obj)
{
    if ( this == obj )
        return true;
```



## 3.5 ObjectBuilder

When you unmarshal an XML document it could happen that you want to compare the resulting Object Model with an expecting Object Model instance.





## 4 Example

In





3/21/2002

-anyURI

-QName

We need now to provide an instance builder i.e. implementing the `ObjectBuilder`.

## 4.2 Creating `PrimitivesBuilder.java`



3/21/2002

```
jar cvf ../PrimitivesWithoutFacetsTest.jar *.*
```

and commit it in the corresponding directory (in this case we will commit  
PrimitivesWithoutFacets.jar in  
castor/src/tests/MasterTestSuite/xml/sourcegenerator.

Then you call `mmibysndi (t followndi:r.) Tj1438 0 TD 0.0456 Tc ( ) Tj ET 0.9529 0.9529 0`



## 5 Status of CTF

The current version of CTF is far from being perfect and clearly needs some improvements.

However it is a solid base to test the behavior of Castor and avoid regression while fixing bugs.

Future versions of the CTF will include:

- Write a tool to generate test acceptance documents.
- Change the architecture of RandomHelper
- Provide



## 6 References

1. Seven Steps to Test Automation Success

Version of July 2000

Bret Pettichord

See [http://www.io.com/~wazmo/papers/seven\\_steps.html](http://www.io.com/~wazmo/papers/seven_steps.html)

2. JUnit Framework

See <http://www.jur ewgml>



3/21/2002

7



3/21/2002



3/21/2002

```
<!-- assume that the Testing Framework will try to match the name -->  
<!--
```





```
<xsd:complexType>
```

```
<xsd:sequence>
```



```
<!--A root object in an object model-->  
<xsd:complexType name="RootType">  
  <xsd:complexContent>  
    <xsd:extension base="StringType">  
      <!--set to true to generate randomly the given Object Model-->  
      <xsd:attribute name="random" type="xsd:boolean" default="false"/>  
    </xsd:extension>  
  </xsd:complexContent>  
</xsd:complexType>
```



3/21/2002

8

## 9 Appendix C: Check-list to write a test

-