Writing Unit Tests

Table of contents

Copyright © 2002-2007 The Apache Software Foundation. All rights reserved.

1 Introduction

Recommended resources

- JUnit homepage (http://www.junit.org)
- JUnit Cookbook (http://junit.sourceforge.net/doc/cookbook/cookbook.htm) (Eric Gamma, Kent Beck)
- JUnit: A Cook's Tour (http://junit.sourceforge.net/doc/cookstour/cookstour.htm) (Eric Gamma, Kent Beck)
- JUnitTest Infected: Programmers Love Writing Tests (http://junit.sourceforge.net/doc/testinfected/testing.htm)

2 Organization

- Put your test classes in *src/test*.
- Add the ant task that executes your test to *src/targets/test-build.xml*.

3 The Test Publication

Most tests will need a publication in the install (servlet container) directory. To provide a predictable test publication, the clean *default* publication from the build directory is copied to the *test* publication in the installation directory.

In the test buildfile, the test publication is setup by the *test.pub.prepare* target. The directory {{${install.dir}/lenya/pubs/test}} is deleted (so that the files created by former tests are removed), and the default publication is copied to this directory. Add this target to the *depends* attribute of your test target if you need the test publication.

4 The PublicationHelper

To simplify the acces to a publication you can use the class *org.apache.lenya.cms.PublicationHelper*. It provides the following methods:

```
/**
 * Initializes the object with the first parameters from the command
 * line arguments <code>args</code>. The remainder of the array is returned.
 * @param args The command line arguments of the test.
 * @return The remainder of the arguments after the publication
 * parameters are extracted.
 */
public static String[] extractPublicationArguments(String args[]);

/**
 * Returns the publication.
 * @return A publication object.
 */
public static Publication getPublication();
```

The *extractPublicationArguments(String[])* method extracts the first two strings from the *args* parameter. The first one is the servlet context path, the second is the publication ID.

To make use of the PublicationHelper, you have to call the *extractPublicationArguments(String[])* method in the *main(String())* method of your *TestCase* class. This initializes the PublicationHelper:

Copyright © 2002-2007 The Apache Software Foundation. All rights reserved.

```
    public static void main(String[] args) {

        // extract the arguments needed for setting up the publication
        // only the remaining arguments are returned
        args = PublicationHelper.extractPublicationArguments(args);

        ...
    }
```

5 A TestCase Skeleton

```
public class MyTest extends TestCase {

    // static fields to store test parameters
    private File configFile;
    ...

    /** Constructor. */
    public MyTest(String test) {
        super(test);
    }

    /**
     * The main program.
     * The parameters are set from the command line arguments.
     *
     * @param args The command line arguments.
     */
    public static void main(String[] args) {
        args = PublicationHelper.extractPublicationArguments(args);
        setConfigFile(args[0]);
        TestRunner.run(getSuite());
    }

    /** Returns the test suite. */
    public static Test getSuite() {
        return new TestSuite(MyTest.class);
    }

    /** Tests whatever you want. */
    public void testSomething() {
        ...
    }

    /** Sets a parameter value. */
    protected static void setConfigFile(String fileName) {
        assertNotNull(string);
        File publicationDirectory
            = PublicationHelper.getPublication().getDirectory();
        configFile = new File(publicationDirectory, fileName);
        assertTrue(configFile.exists());
    }

    /** Returns a parameter value. */
    protected static File getConfigFile() {
        return configFile;
    }
}
```

Copyright © 2002-2007 The Apache Software Foundation. All rights reserved.

6 Debugging a Test

For debugging, it might be desired to run the test from an API. In this case, the *main(String[])* method is never executed.

To provide the parameters, you can hardcode them as fallback in the TestCase.setup() method that is called before the test is invoked:

```
    /** @see junit.framework.TestCase#setUp() */
    protected void setUp() throws Exception {
        if (getConfigFile() == null) {
            String args[] = {
                "D:\\Development\\build\\tomcat-4.1.24\\webapps\\lenya",
                "test"
            };
            PublicationHelper.extractPublicationArguments(args);
            setConfigFile("config/something.xconf");
        }
    }
```

7 The Test Buildfile

The test buildfile is located at *src/targets/test-build.xml*. It contains the following common targets:

- **test** - Runs all tests.
- **tests.junit** - Runs the JUnit tests.
- **tests.anteater** - Runs the Anteater tests.
- **tests.prepare** - Prepares the tests, e.g. compiles test classes.
- **test.pub.prepare** - Prepares the test publication.

8 Adding the Test to the Buildfile

To add your test to the buildfile, you create a target called *test.<name>*.

If you use assertions (Java assertions, not the JUnit ones) in your test, it is important to enable them using the *-ea* or *-enableassertions* argument.

```
  <target name="test.my" depends="test.pub.prepare">
    <!-- My Test -->
    <java fork="yes" classname="org.apache.lenya.cms.mypackage.MyTest">
      <jvmarg value="-enableassertions"/>

      <arg value="${install.dir}"/>           // PublicationHelper
      <arg value="test"/>                     // PublicationHelper
      <arg value="config/something.xconf"/>   // MyTest

      <classpath refid="classpath"/>

      <classpath>
        <pathelement location="${build.test}" />
        <pathelement path="${build.root}/lenya/webapp/WEB-INF/classes" />
        <fileset dir="${build.root}/lenya/webapp/WEB-INF/lib">
          <include name="ant**.jar"/>
        </fileset>

      </classpath>
    </java>
```

Copyright © 2002-2007 The Apache Software Foundation. All rights reserved.

```
   </target>
```

Finally, you have to add the test to the *tests.junit* target:

```


<target name="tests.junit" depends="init, tests.prepare, ..., test.my">
```

Now you can run the tests:

```

$LENYA_HOME > build test
```

If you want to call your test independently, you have to call the preparation targets before:

```

$LENYA_HOME > build init
$LENYA_HOME > build tests.prepare
$LENYA_HOME > build test.my
```

Copyright © 2002-2007 The Apache Software Foundation. All rights reserved.