TestingVelocity

I'm sure many people want to test their Velocity templates. This is a page for such ideas.

TODO: add ideas 😉

Approach 1: The obvious approach: Run and check.

Run Velocity against the template to be checked. Look at the output and see if it is what you desired. This must be the most primitive testing approach, but most people nowadays are too lazy and want this done by the computer. The rest of this page will discuss automated testing approaches.

Approach 2: The Golden File.

This is how the Velocity regression tests work. (Or at least how I think it works. Correct me if I'm wrong.) The first part is like Approach 1. You run the template and check that it is right. Once you know it's correct, save the correct output (called the Golden File) and have JUnit compare the template output and the Golden File. Of course, it will pass. What's more, it will break whenever you screw up the template. (The Velocity tests will break whenever somebody screws up Velocity itself.)

Check out the test source code for more on how this is done.

Approach 3: VTL snippets.

Instead of testing the whole template, I came up with a way to test template snippets. This was more or less used as learning tests. There are three classes used: VelocityTestTool, AbstractVelocityTestCase and AbstractVelocityMockStrutsTestCase. Developers will subclass AbstractVelocity*TestCase and do the usuall JUnit assertion like this:

```
import org.apache.velocity.app.Velocity;
import org.apache.velocity.app.VelocityEngine;
import org.apache.velocity.runtime.RuntimeConstants;

public class EscapeToolTest extends AbstractVelocityTestCase
{
    public void testStringJava() throws Exception
    {
        this.getContext().put("ctx", this.getContext());
        this.getContext().put("escape", new EscapeTool());
        this.getContext().put("java", "He didn't say, \"Stop!\"");

        this.setTemplate("" +
                "$escape.java($java)\n" +
                "");

        this.setExpected("" +
                "He didn't say, \\\"Stop!\\\"\n" +
                "");

        this.assertVelocity();
    }
}
```

You may have noticed that the snippets are set as java String objects, so there is no need for template files. (But you will need to escape special java characters.) A bit more information can be found here.

Here is what each class looks like:

- VelocityTestTool.java

```
import org.apache.velocity.app.Velocity;
import org.apache.velocity.app.VelocityEngine;
import org.apache.velocity.runtime.RuntimeConstants;
import org.apache.velocity.tools.generic.log.CommonsLogLogSystem;

public class VelocityTestTool
{

    public static VelocityEngine newEngine()
    {
        VelocityEngine engine = new VelocityEngine();

        engine.setProperty(RuntimeConstants.RUNTIME_LOG_LOGSYSTEM_CLASS, CommonsLogLogSystem.class.getName());
        engine.setProperty("runtime.log.logsystem.commons.logging.name", "test.velocity.log");

        return engine;
    }

    public static void initSingletonLogger()
    {
        Velocity.setProperty(RuntimeConstants.RUNTIME_LOG_LOGSYSTEM_CLASS, CommonsLogLogSystem.class.getName());
        Velocity.setProperty("runtime.log.logsystem.commons.logging.name", "test.velocity.log");
    }

}
```

- AbstractVelocityTestCase.java

```
import java.io.IOException;
import java.io.StringWriter;

import junit.framework.TestCase;

import org.apache.commons.logging.Log;
import org.apache.commons.logging.LogFactory;
import org.apache.velocity.VelocityContext;
import org.apache.velocity.app.VelocityEngine;
import org.apache.velocity.context.Context;
import org.apache.velocity.exception.MethodInvocationException;
import org.apache.velocity.exception.ParseErrorException;
import org.apache.velocity.exception.ResourceNotFoundException;

public abstract class AbstractVelocityTestCase extends TestCase
{

    private Log log = LogFactory.getLog(this.getClass());
    protected final Log getLog()
    {
        return this.log;
    }

    /* (non-Javadoc)
     * @see junit.framework.TestCase#setUp()
     */
    protected void setUp() throws Exception
    {
        super.setUp();

        this.setEngine(VelocityTestTool.newEngine());
        this.setTemplate("");
        this.setContext(new VelocityContext());
        this.setExpected("");
    }

    /* (non-Javadoc)
     * @see junit.framework.TestCase#tearDown()
     */
    protected void tearDown() throws Exception
```

```
{
    this.setEngine(null);
    this.setTemplate(null);
    this.setContext(null);
    this.setExpected(null);

    super.tearDown();
}

private VelocityEngine engine = null;
private String template = "";
private Context context = null;
private String expected = null;

/**
 * @return Returns the engine.
 */
protected VelocityEngine getEngine()
{
    return this.engine;
}
/**
 * @param engine The engine to set.
 */
protected void setEngine(VelocityEngine engine)
{
    this.engine = engine;
}
/**
 * @return Returns the template.
 */
protected String getTemplate()
{
    return this.template;
}
/**
 * @param template The template to set.
 */
protected void setTemplate(String template)
{
    this.template = template;
}
/**
 * @return Returns the context.
 */
protected Context getContext()
{
    return this.context;
}
/**
 * @param context The context to set.
 */
protected void setContext(Context context)
{
    this.context = context;
}
/**
 * @return Returns the expected.
 */
protected String getExpected()
{
    return this.expected;
}
/**
 * @param expected The expected to set.
 */
protected void setExpected(String expected)
{
    this.expected = expected;
}
```

```
    protected void assertVelocity() throws ParseErrorException, MethodInvocationException,
ResourceNotFoundException, IOException, Exception
    {
        this.getEngine().init();

        StringWriter writer = new StringWriter();
        assertTrue(this.getEngine().evaluate(this.getContext(), writer, this.getName(), this.getTemplate()));

        assertEquals(this.getExpected(), String.valueOf(writer));
    }
}
```

- AbstractVelocityMockStrutsTestCase.java

```
import java.io.IOException;
import java.io.StringWriter;

import org.apache.commons.logging.Log;
import org.apache.commons.logging.LogFactory;
import org.apache.velocity.VelocityContext;
import org.apache.velocity.app.VelocityEngine;
import org.apache.velocity.context.Context;
import org.apache.velocity.exception.MethodInvocationException;
import org.apache.velocity.exception.ParseErrorException;
import org.apache.velocity.exception.ResourceNotFoundException;
import org.apache.velocity.tools.view.context.ChainedContext;

import servletunit.struts.MockStrutsTestCase;

public abstract class AbstractVelocityMockStrutsTestCase extends MockStrutsTestCase
{

    private Log log = LogFactory.getLog(this.getClass());
    protected final Log getLog()
    {
        return this.log;
    }

    /* (non-Javadoc)
     * @see junit.framework.TestCase#setUp()
     */
    protected void setUp() throws Exception
    {
        super.setUp();

        this.setEngine(VelocityTestTool.newEngine());
        this.setTemplate("");
        this.setContext(new ChainedContext(new VelocityContext(), this.getEngine(), this.getRequest(), this.
getResponse(), this.getActionServlet().getServletContext()));
        this.setExpected("");
    }

    /* (non-Javadoc)
     * @see junit.framework.TestCase#tearDown()
     */
    protected void tearDown() throws Exception
    {
        this.setEngine(null);
        this.setTemplate(null);
        this.setContext(null);
        this.setExpected(null);

        super.tearDown();
    }

    private VelocityEngine engine = null;
    private String template = "";
    private Context context = null;
    private String expected = null;
```

```
    /**
     * @return Returns the engine.
     */
    protected VelocityEngine getEngine()
    {
        return this.engine;
    }
    /**
     * @param engine The engine to set.
     */
    protected void setEngine(VelocityEngine engine)
    {
        this.engine = engine;
    }
    /**
     * @return Returns the template.
     */
    protected String getTemplate()
    {
        return this.template;
    }
    /**
     * @param template The template to set.
     */
    protected void setTemplate(String template)
    {
        this.template = template;
    }
    /**
     * @return Returns the context.
     */
    protected Context getContext()
    {
        return this.context;
    }
    /**
     * @param context The context to set.
     */
    protected void setContext(Context context)
    {
        this.context = context;
    }
    /**
     * @return Returns the expected.
     */
    protected String getExpected()
    {
        return this.expected;
    }
    /**
     * @param expected The expected to set.
     */
    protected void setExpected(String expected)
    {
        this.expected = expected;
    }

    protected void assertVelocity() throws ParseErrorException, MethodInvocationException,
ResourceNotFoundException, IOException, Exception
    {
        this.getEngine().init();

        StringWriter writer = new StringWriter();
        assertTrue(this.getEngine().evaluate(this.getContext(), writer, this.getName(), this.getTemplate()));

        assertEquals(this.getExpected(), String.valueOf(writer));
    }
}
```

Approach 4: Cactus integeration.

You can test Velocity templates in your web application by integerating Cactus with VelocityViewServlet.

If your template at `/test/test.vm` looks like this:

```
<html>
  <body>
    Hello $user !
  </body>
</html>
```

Then your TestCase might look like this:

```
import org.apache.cactus.ServletTestCase;
import org.apache.cactus.WebRequest;
import org.apache.cactus.WebResponse;
import org.apache.velocity.tools.view.servlet.VelocityViewServlet;

public class VelocityTest extends ServletTestCase
{

    public void beginTestVelocity(WebRequest theRequest)
    {
        theRequest.setURL("localhost:8080", "/CactusDemo", "/test/test.vm", null, null);
    }

    public void testTestVelocity() throws Exception
    {
        VelocityViewServlet servlet = new VelocityViewServlet();
        servlet.init(this.config);

        this.request.setAttribute("user", "Servlet User from Cactus");

        servlet.doGet(this.request, this.response);
    }

    public void endTestVelocity(WebResponse theResponse)
    {
        String expected = "" +
                "<html>\n" +
                "  <body>\n" +
                "    Hello Servlet User from Cactus !\n" +
                "  </body>\n" +
                "</html>\n" +
                "";

        assertEquals(expected, theResponse.getText());
    }
}
```

This assumes that you are running your web application at `http://localhost:8080/CactusDemo/`.