SilentParseInclude

Fred Toth came up with a great idea on a "silent" #parse/#include. Unfortunately, it uses the SingletonModel, which means that it does not work with the VelocityViewServlet in VelocityTools 1.2. Here is the same solution written for 1.2.

Fred Toth's original solution can be found here.

- A mini "tool" for the tool box:

```
package org.apache.velocity.tools.generic;

import org.apache.velocity.app.Velocity;
import org.apache.velocity.app.VelocityEngine;
import org.apache.velocity.tools.view.context.ViewContext;
import org.apache.velocity.tools.view.tools.ViewTool;

public class TemplateExistsTool implements ViewTool
{

    private VelocityEngine engine = null;

    public VelocityEngine getVelocityEngine()
    {
        return this.engine;
    }

    public void setVelocityEngine(VelocityEngine engine)
    {
        this.engine = engine;
    }

    public void init(Object obj)
    {
        // if ViewContext, try to retrieve a VelocityEngine for use
        if (obj instanceof ViewContext)
        {
            this.setVelocityEngine(((ViewContext) obj).getVelocityEngine());
        }
    }

    public boolean exists(String template)
    {
        if (this.engine == null)
        {
            return Velocity.resourceExists(template);
        }

        return this.engine.templateExists(template);
    }

}
```

- A tool box entry:

```
<tool>
  <key>ftest</key>
  <scope>request</scope>
  <class>org.apache.velocity.tools.generic.TemplateExistsTool</class>
</tool>
```

- And a couple of macro definitions:

```
#macro( cinclude $file )##
#if($ftest.exists($file))#include($file)#end##
#end
#macro( cparse $file )##
#if($ftest.exists($file))#parse($file)#end##
#end
```

With the above you can use these macros to get the job done:

```
#cinclude("file_that_may_not_be_there")
#cparse("another_file_that_may_not_be_there")
```

If the files are not there, the macro silently evaluates to the empty string. If the files exist, you get the normal behavior.