VelocityMail

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

import java.io.IOException;
import java.io.InputStream;
import java.io.StringWriter;
import java.util.Enumeration;
import java.util.HashMap;
import java.util.Map;
import java.util.Properties;

import javax.mail.BodyPart;
import javax.mail.Message;
import javax.mail.Multipart;
import javax.mail.Session;
import javax.mail.Transport;
import javax.mail.internet.InternetAddress;
import javax.mail.internet.MimeBodyPart;
import javax.mail.internet.MimeMessage;
import javax.mail.internet.MimeMultipart;
import javax.swing.border.EtchedBorder;

import org.apache.commons.collections.ExtendedProperties;
import org.apache.velocity.Template;
import org.apache.velocity.VelocityContext;
import org.apache.velocity.app.Velocity;
import org.apache.velocity.context.Context;

/**
 * This class is used to execute velocity templates and send the result via
 * email.
 * <p/>
 * <b>Usage: <b><br/>
 * To control the operation of VelocityMail you need to create a configuration file
 * called 'velocity.mail.properties'. The file MUST
 * be on your class path.
 * <i>Config file </i><br/>
 * <pre>
 *
 *       ## TEST CONFIGURATION
 *       #From Name &amp; Address
 *       #The values of these properties are also velocity templates
 *       test.from.name=$name
 *       test.from.address=test@localhost
 *
 *       #Subject
 *       #Can contain velocity template language
 *       test.subject=Test to $name
 *
 *       #SMTP Host
 *       test.host=127.0.0.1
 *
 *       #Different Mime type templates
 *       test.message.text/plain=/mail/test.text.vm
 *       test.message.text/html=/mail/test.html.vm
 *
 * </pre>
 *
 * <p/>
 * <i>Source Coude </i> <br/>
 * <pre>
 * context.put(&quot;name&quot;, &quot;John Doe&quot;);
 * VelocityMail vmail = VelocityMail.getDefaultInstance();
 * InternetAddress address = new InternetAddress(&quot;test@localhost&quot;);
 * vmail.send(&quot;test&quot;, address, context);
 * </pre>
 *
```

```
 * @author <a href="mailto:leon@psybergate.com">Leon Messerschmidt </a>
 * @version $revision$
 *
 */
public class VelocityMail
{

    /**
     * The default instance for the Velocity Mail class
     */
    private static VelocityMail defaultInstance;

    /**
     * The configuration for Velocity Mail
     */
    private ExtendedProperties properties;

    /**
     * The list of email configurations that are available.
     */
    private Map configurations;

    /**
     * Create a new Velocity Mail object with a properties object
     *
     * @param properties
     */
    public VelocityMail(ExtendedProperties properties)
    {
        this.configurations = new HashMap();
        this.properties = properties;
    }

    /**
     * Return the default instance for VelocityMail. It attempts to load the
     * resources "velocity.mail.properties" from the classpath.
     *
     * @return @throws
     *         IOException
     */
    public static VelocityMail getDefaultInstance() throws IOException
    {
        if (defaultInstance == null)
        {
            ClassLoader classLoader = VelocityMail.class.getClassLoader();
            InputStream inputStream = classLoader
                    .getResourceAsStream("velocity.mail.properties");
            ExtendedProperties extendedProperties = new ExtendedProperties();
            extendedProperties.load(inputStream);
            defaultInstance = new VelocityMail(extendedProperties);
        }
        return defaultInstance;
    }

    /**
     * Send an email message to a single address using the default
     * configuration.
     *
     * @param address
     *            The receiver of the email
     * @param context
     *            The velocity context used to merge the email template
     *
     * @throws Exception
     */
    public void send(InternetAddress address, Context context) throws Exception
    {
        InternetAddress[] addresses = new InternetAddress[1];
        addresses[0] = address;
        send(addresses, context);
    }
```

```
/**
 * Send an email message to a single address
 *
 * @param config
 *            The configuration to use (specified in the email properties)
 * @param address
 *            The receiver of the email
 * @param context
 *            The velocity context used to merge the email template
 * @throws Exception
 */
public void send(String config, InternetAddress address, Context context)
        throws Exception
{
    InternetAddress[] addresses = new InternetAddress[1];
    addresses[0] = address;
    send(config, addresses, context);
}

/**
 * Send an email message to a list of addresses using the default
 * configuration.
 *
 * @param address
 *            The receivers of the email
 * @param context
 *            The velocity context used to merge the email template
 *
 * @throws Exception
 */
public void send(InternetAddress[] addresses, Context context)
        throws Exception
{
    send("default", addresses, context);
}

/**
 * Send an email message to a list of addresses
 *
 * @param config
 *            The configuration to use (specified in the email properties)
 * @param address
 *            The receivers of the email
 * @param context
 *            The velocity context used to merge the email template
 * @throws Exception
 */
public void send(String config, InternetAddress[] addresses, Context context)
        throws Exception
{

    ExtendedProperties configuration = getConfiguration(config);

    Message message = createMessage(configuration, context);
    setMessageContent(message, configuration, context);
    message.setRecipients(Message.RecipientType.TO, addresses);

    Transport.send(message);

}

/**
 * Create a JavaMail message object from a specified configuration
 *
 * @param configuration
 *            A configuration object that was created from VelocityMail
 *            properties
 *
 * @param context
 *            The velocity context to use to merge the email template
```

```
     *
     * @throws Exception
     */
    protected Message createMessage(ExtendedProperties configuration,
            Context context) throws Exception
    {
        String host = configuration.getString("host");
        Properties properties = new Properties();
        properties.put("mail.smtp.host", host);
        Session session = Session.getInstance(properties);

        MimeMessage result = new MimeMessage(session);

        String subject = evaluate(configuration, "subject", context);
        result.setSubject(subject);

        String fromAddr = evaluate(configuration, "from.address", context);
        String fromName = evaluate(configuration, "from.name", context);
        InternetAddress fromAddress = new InternetAddress(fromAddr, fromName);
        result.setFrom(fromAddress);

        return result;
    }

    /**
     * Merge the email template(s) using the velocity context and add the
     * content to the JavaMail message object
     *
     * @param message
     *            The message that will receive the message content
     *
     * @param configuration
     *            The configuration to use
     *
     * @param context
     *            The velocity context that will be used to merge the mail
     *            templates
     *
     * @throws Exception
     */
    protected void setMessageContent(Message message,
            ExtendedProperties configuration, Context context) throws Exception
    {
        ExtendedProperties content = configuration.subset("message");
        Enumeration keys = content.keys();

        Multipart multipart = new MimeMultipart();

        while (keys.hasMoreElements())
        {
            String key = (String) keys.nextElement();
            String templateName = content.getString(key);
            Template template = Velocity.getTemplate(templateName);
            StringWriter writer = new StringWriter();
            template.merge(context, writer);
            writer.close();
            BodyPart bodyPart = new MimeBodyPart();
            bodyPart.setContent(writer.toString(), key);
            multipart.addBodyPart(bodyPart);
        }

        message.setContent(multipart);
    }

    /**
     * Helper method to evaluate a string
     *
     * @param configuration
     * @param property
     * @param context
     * @return @throws
```

```
     *         Exception
     */
    protected String evaluate(ExtendedProperties configuration,
            String property, Context context) throws Exception
    {
        Velocity.init();

        String subjectTemplate = configuration.getString(property);
        StringWriter stringWriter = new StringWriter();
        Velocity.evaluate(context, stringWriter, "email", subjectTemplate);
        stringWriter.close();
        return stringWriter.toString();
    }

    /**
     * Get the configuration for a given configuration name
     *
     * @param name
     * @return
     */
    protected ExtendedProperties getConfiguration(String name)
    {
        ExtendedProperties config = (ExtendedProperties) configurations
                .get(name);
        if (config == null)
        {
            config = properties.subset(name);
            configurations.put(name, config);
        }
        return config;
    }

}
```