ListTool

Replacement of ArrayTool.

Separated the tool into two.

ListTool (basic functions for manipulating lists)

- get/set/size support
- isEmpty support
- contains support
- isList/isArray

Extended ListTool (extended functions that we are not sure we need)

- conversion between Lists and arrays
- clone support
- length support (same as size)

The ListTool is now kept in the SVN repo! The Extended ListTool is still below for those interested in it. 🙂

```
/*
 * Copyright 2003-2005 The Apache Software Foundation.
 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 *     http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */

package org.apache.velocity.tools.generic;

import java.lang.reflect.Array;
import java.lang.reflect.Method;
import java.util.ArrayList;
import java.util.List;

/**
 * Tool for working with Lists and arrays in Velocity templates.
 * In addition to the features ListTool provides,
 * it provides the function to retrieve the length,
 * and create clones of a list or an array object.
 * Also provides a method to convert arrays into Lists and
 * Lists into arrays.
 *
 * <p><pre>
 * Example uses:
 *  $primes                  -> new int[] {2, 3, 5, 7}
 *  $list.length($primes)    -> 4
 *  $list.get($primes, 2)    -> 5
 *  $list.clone($primes)     -> int[] {2, 3, 5, 7}, != $primes
 *  $list.set($primes, 2, 1) -> (primes[2] becomes 1)
 *  $list.get($primes, 2)    -> 1
 *  $list.get($clone, 2)     -> 5
 *
 * Example toolbox.xml config (if you want to use this with VelocityView):
 * &lt;tool&gt;
 *   &lt;key&gt;list&lt;/key&gt;
 *   &lt;scope&gt;application&lt;/scope&gt;
 *   &lt;class&gt;org.apache.velocity.tools.generic.ExtendedListTool&lt;/class&gt;
 * &lt;/tool&gt;
 * </pre></p>
 *
 * <p>This tool is entirely threadsafe, and has no instance members.
 * It may be used in any scope (request, session, or application).
```

```
 * </p>
 *
 * @author <a href="mailto:shinobu@ieee.org">Shinobu Kawai</a>
 * @version $Id: $
 */
public class ExtendedListTool extends ListTool
{

    /**
     * Default constructor.
     */
    public ExtendedListTool()
    {
    }

    /**
     * Converts an array object into a List.
     * <ul>
     *   <li>
     *     If the object is already a List,
     *     it will return the object itself.
     *   </li>
     *   <li>
     *     If the object is an array of an Object,
     *     it will return a List of the elements.
     *   </li>
     *   <li>
     *     If the object is an array of a primitive type,
     *     it will return a List of the elements wrapped in their wrapper class.
     *   </li>
     *   <li>
     *     If the object is none of the above, it will return null.
     *   </li>
     * </ul>
     * @param array an array object.
     * @return the converted java.util.List.
     */
    public List toList(Object array)
    {
        if (this.isList(array)) {
            return (List) array;
        }
        if (!this.isArray(array))
        {
            return null;
        }

        // Thanks to Eric Fixler for this refactor.
        int length = Array.getLength(array);
        List asList = new ArrayList(length);
        for (int index = 0; index < length; ++index)
        {
            asList.add(Array.get(array, index));
        }
        return asList;
    }

    /**
     * Converts a List object into an array.
     * <ul>
     *   <li>
     *     If the object is already an array,
     *     it will return the object itself.
     *   </li>
     *   <li>
     *     If the object is a List,
     *     it will return an array according to the object's
     *     <code>toArray()</code> method.
     *   </li>
     *   <li>
     *     If the object is none of the above, it will return null.
```

```
 *   </li>
 * </ul>
 * @param list a List object.
 * @return the converted array.
 */
public Object toArray(Object list)
{
    if (this.isArray(list))
    {
        return list;
    }
    if (!this.isList(list))
    {
        return null;
    }

    List asList = (List) list;
    return asList.toArray(new Object[asList.size()]);
}

/**
 * Gets the length of an array/List.
 * It will return null under the following conditions:
 * <ul>
 *   <li><code>array</code> is null.</li>
 *   <li><code>array</code> is not an array/List.</li>
 * </ul>
 * The result will be same as the {@link #size(Object)} method.
 * @param array the array object.
 * @return the length of the array.
 * @see #size(Object)
 */
public Integer length(Object array)
{
    if (this.isList(array))
    {
        return this.size(array);
    }
    if (!this.isArray(array))
    {
        return null;
    }

    // Thanks to Eric Fixler for this refactor.
    return new Integer(Array.getLength(array));
}

/**
 * Gets the clone of a List/array.
 * It will return null under the following conditions:
 * <ul>
 *   <li><code>list</code> is null.</li>
 *   <li><code>list</code> is not a List/array.</li>
 * </ul>
 * @param list the List/array object.
 * @return the clone of the List/array.
 */
public Object clone(Object list)
{
    if (this.isArray(list))
    {
        Class type = list.getClass().getComponentType();
        int length = Array.getLength(list);
        Object clone = Array.newInstance(type, length);
        System.arraycopy(list, 0, clone, 0, length);
        return clone;
    }
    if (!this.isList(list))
    {
        return null;
    }
```

```
        // first, try the clone() method.
        Class clazz = list.getClass();
        try
        {
            Method cloneMethod = clazz.getMethod("clone", new Class[0]);
            return cloneMethod.invoke(list, null);
        }
        catch (Exception ignoreAndTryTheNextStep)
        {
        }

        // try to copy to a new instance.
        try
        {
            List clone = (List) clazz.newInstance();
            clone.addAll(((List) list));
            return clone;
        }
        catch (Exception ignoreAndTryTheNextStep)
        {
        }

        // last resort.
        return new ArrayList(((List) list));
    }

}
```