HTMLTransfer.java - RTFTransfer.java

```
/*******************************************************************************
 * Copyright (c) 2000, 2008 IBM Corporation and others.
 * All rights reserved. This program and the accompanying materials
 * are made available under the terms of the Eclipse Public License v1.0
 * which accompanies this distribution, and is available at
 * http://www.eclipse.org/legal/epl-v10.html
 *
 * Contributors:
 *     IBM Corporation - initial API and implementation
 *******************************************************************************/
package org.eclipse.swt.dnd;


/**
 * The class <code>HTMLTransfer</code> provides a platform specific mechanism
 * for converting text in HTML format represented as a java <code>String</code>
 * to a platform specific representation of the data and vice versa.
 *
 * <p>An example of a java <code>String</code> containing HTML text is shown
 * below:</p>
 *
 * <code><pre>
 *     String htmlData = "<p>This is a paragraph of text.</p>";
 * </code></pre>
 *
 * @see Transfer
 */
public class HTMLTransfer extends ByteArrayTransfer {

    private static HTMLTransfer _instance = new HTMLTransfer();
    private static final String TYPENAME1 = "text/html¥0";
    private static final int TYPEID1 = registerType(TYPENAME1);
    private static final String TYPENAME2 = "TEXT/HTML¥0";
    private static final int TYPEID2 = registerType(TYPENAME2);


private HTMLTransfer() {
}
/**
 * Returns the singleton instance of the HTMLTransfer class.
 *
 * @return the singleton instance of the HTMLTransfer class
 */
public static HTMLTransfer getInstance () {
    return _instance;
}
/**
 * This implementation of <code>javaToNative</code> converts HTML-formatted text
 * represented by a java <code>String</code> to a platform specific representation.
 *
 * @param object a java <code>String</code> containing HTML text
 * @param transferData an empty <code>TransferData</code> object that will
 *      be filled in on return with the platform specific format of the data
 *
```

```
/*******************************************************************************
 * Copyright (c) 2000, 2008 IBM Corporation and others.
 * All rights reserved. This program and the accompanying materials
 * are made available under the terms of the Eclipse Public License v1.0
 * which accompanies this distribution, and is available at
 * http://www.eclipse.org/legal/epl-v10.html
 *
 * Contributors:
 *     IBM Corporation - initial API and implementation
 *******************************************************************************/
package org.eclipse.swt.dnd;


/**
 * The class <code>RTFTransfer</code> provides a platform specific mechanism
 * for converting text in RTF format represented as a java <code>String</code>
 * to a platform specific representation of the data and vice versa.
 *
 * <p>An example of a java <code>String</code> containing RTF text is shown
 * below:</p>
 *
 * <code><pre>
 *     String rtfData = "{¥¥rtf1{¥¥colortbl;¥¥red255¥¥green0¥¥blue0;}¥¥uc1¥¥b¥¥i
 Hello World}";
 * </code></pre>
 *
 * @see Transfer
 */
public class RTFTransfer extends ByteArrayTransfer {

    private static RTFTransfer _instance = new RTFTransfer();
    private static final String TYPENAME1 = "text/rtf¥0";
    private static final int TYPEID1 = registerType(TYPENAME1);
    private static final String TYPENAME2 = "TEXT/RTF¥0";
    private static final int TYPEID2 = registerType(TYPENAME2);
    private static final String TYPENAME3 = "application/rtf¥0";
    private static final int TYPEID3 = registerType(TYPENAME3);

private RTFTransfer() {
}
/**
 * Returns the singleton instance of the RTFTransfer class.
 *
 * @return the singleton instance of the RTFTransfer class
 */
public static RTFTransfer getInstance () {
    return _instance;
}
/**
 * This implementation of <code>javaToNative</code> converts RTF-formatted text
 * represented by a java <code>String</code> to a platform specific representation.
 *
 * @param object a java <code>String</code> containing RTF text
 * @param transferData an empty <code>TransferData</code> object that will
 *      be filled in on return with the platform specific format of the data
 *
```

HTMLTransfer.java - RTFTransfer.java

```
 * @see Transfer#nativeToJava
 */
public void javaToNative (Object object, TransferData transferData){
}
/**
 * This implementation of <code>nativeToJava</code> converts a platform specific
 * representation of HTML text to a java <code>String</code>.
 *
 * @param transferData the platform specific representation of the data to be
 converted
 * @return a java <code>String</code> containing HTML text if the conversion was
 successful;
 *      otherwise null
 *
 * @see Transfer#javaToNative
 */
public Object nativeToJava(TransferData transferData){
    return null;
}
protected String[] getTypeNames(){
    return new String[]{TYPENAME1, TYPENAME2};
}
protected int[] getTypeIds(){
    return new int[]{TYPEID1, TYPEID2};
}
}

```

```
 * @see Transfer#nativeToJava
 */
public void javaToNative (Object object, TransferData transferData){
}
/**
 * This implementation of <code>nativeToJava</code> converts a platform specific
 * representation of RTF text to a java <code>String</code>.
 *
 * @param transferData the platform specific representation of the data to be
 converted
 * @return a java <code>String</code> containing RTF text if the conversion was
 successful;
 *      otherwise null
 *
 * @see Transfer#javaToNative
 */
public Object nativeToJava(TransferData transferData){
    return null;
}
protected String[] getTypeNames(){
    return new String[]{TYPENAME1, TYPENAME2, TYPENAME3};
}
protected int[] getTypeIds(){
    return new int[]{TYPEID1, TYPEID2, TYPEID3};
}
}

```