

## HTMLTransfer.java – RTFTransfer.java

```

1/*****
2 * Copyright (c) 2000, 2008 IBM Corporation and others.
3 * All rights reserved. This program and the accompanying materials
4 * are made available under the terms of the Eclipse Public License v1.0
5 * which accompanies this distribution, and is available at
6 * http://www.eclipse.org/legal/epl-v10.html
7 *
8 * Contributors:
9 *   IBM Corporation - initial API and implementation
10 *****/
11 package org.eclipse.swt.dnd;
12
13 /**
14 * The class <code>HTMLTransfer</code> provides a platform specific mechanism
15 * for converting text in HTML format represented as a java <code>String</code>
16 * to a platform specific representation of the data and vice versa.
17 *
18 * <p>An example of a java <code>String</code> containing HTML text is shown
19 * below:</p>
20 *
21 * <code><pre>
22 *   String htmlData = "<p>This is a paragraph of text.</p>";
23 * </code></pre>
24 *
25 * @see Transfer
26 */
27 public class HTMLTransfer extends ByteArrayTransfer {
28
29     private static HTMLTransfer _instance = new HTMLTransfer();
30     private static final String TYPENAME1 = "text/html¥0";
31     private static final int TYPEID1 = registerType(TYPENAME1);
32     private static final String TYPENAME2 = "TEXT/HTML¥0";
33     private static final int TYPEID2 = registerType(TYPENAME2);
34
35     private HTMLTransfer() {
36     }
37
38     /**
39     * Returns the singleton instance of the HTMLTransfer class.
40     * @return the singleton instance of the HTMLTransfer class
41     */
42     public static HTMLTransfer getInstance () {
43         return _instance;
44     }
45
46     /**
47     * This implementation of <code>javaToNative</code> converts HTML-formatted text
48     * represented by a java <code>String</code> to a platform specific representation.
49     * @param object a java <code>String</code> containing HTML text
50     * @param transferData an empty <code>TransferData</code> object that will
51     *     be filled in on return with the platform specific format of the data
52     */

```

```

1/*****
2 * Copyright (c) 2000, 2008 IBM Corporation and others.
3 * All rights reserved. This program and the accompanying materials
4 * are made available under the terms of the Eclipse Public License v1.0
5 * which accompanies this distribution, and is available at
6 * http://www.eclipse.org/legal/epl-v10.html
7 *
8 * Contributors:
9 *   IBM Corporation - initial API and implementation
10 *****/
11 package org.eclipse.swt.dnd;
12
13 /**
14 * The class <code>RTFTransfer</code> provides a platform specific mechanism
15 * for converting text in RTF format represented as a java <code>String</code>
16 * to a platform specific representation of the data and vice versa.
17 *
18 * <p>An example of a java <code>String</code> containing RTF text is shown
19 * below:</p>
20 *
21 * <code><pre>
22 *   String rtfData = "{¥¥rtf1{¥¥colortbl;¥¥red255¥¥green0¥¥blue0;}¥¥uc1¥¥b¥¥i
23 *   Hello World}";
24 * </code></pre>
25 *
26 * @see Transfer
27 */
28 public class RTFTransfer extends ByteArrayTransfer {
29
30     private static RTFTransfer _instance = new RTFTransfer();
31     private static final String TYPENAME1 = "text/rtf¥0";
32     private static final int TYPEID1 = registerType(TYPENAME1);
33     private static final String TYPENAME2 = "TEXT/RTF¥0";
34     private static final int TYPEID2 = registerType(TYPENAME2);
35     private static final String TYPENAME3 = "application/rtf¥0";
36     private static final int TYPEID3 = registerType(TYPENAME3);
37
38     private RTFTransfer() {
39     }
40
41     /**
42     * Returns the singleton instance of the RTFTransfer class.
43     * @return the singleton instance of the RTFTransfer class
44     */
45     public static RTFTransfer getInstance () {
46         return _instance;
47     }
48
49     /**
50     * This implementation of <code>javaToNative</code> converts RTF-formatted text
51     * represented by a java <code>String</code> to a platform specific representation.
52     * @param object a java <code>String</code> containing RTF text
53     * @param transferData an empty <code>TransferData</code> object that will
54     *     be filled in on return with the platform specific format of the data
55     */

```

## HTMLTransfer.java - RTFTransfer.java

```

53 * @see Transfer#nativeToJava
54 */
55 public void javaToNative (Object object, TransferData transferData) {
56 }
57 /**
58 * This implementation of <code>nativeToJava</code> converts a platform specific
59 * representation of HTML text to a java <code>String</code>.
60 *
61 * @param transferData the platform specific representation of the data to be
    converted
62 * @return a java <code>String</code> containing HTML text if the conversion was
    successful;
63 *     otherwise null
64 *
65 * @see Transfer#javaToNative
66 */
67 public Object nativeToJava(TransferData transferData) {
68     return null;
69 }
70 protected String[] getTypeNames() {
71     return new String[] {TYPENAME1, TYPENAME2};
72 }
73 protected int[] getTypeIds() {
74     return new int[] {TYPEID1, TYPEID2};
75 }
76 }
77

```

```

56 * @see Transfer#nativeToJava
57 */
58 public void javaToNative (Object object, TransferData transferData) {
59 }
60 /**
61 * This implementation of <code>nativeToJava</code> converts a platform specific
62 * representation of RTF text to a java <code>String</code>.
63 *
64 * @param transferData the platform specific representation of the data to be
    converted
65 * @return a java <code>String</code> containing RTF text if the conversion was
    successful;
66 *     otherwise null
67 *
68 * @see Transfer#javaToNative
69 */
70 public Object nativeToJava(TransferData transferData) {
71     return null;
72 }
73 protected String[] getTypeNames() {
74     return new String[] {TYPENAME1, TYPENAME2, TYPENAME3};
75 }
76 protected int[] getTypeIds() {
77     return new int[] {TYPEID1, TYPEID2, TYPEID3};
78 }
79 }
80

```