HOST SCREEN COLOR EXTRACTION IN HATS

Sometimes it is desired to insert text components from a host screen on the web page transformation such that the original color of the components is retained from the host screen. For example if on a host screen, some labels may be displayed in green and some in red. We'll discuss in this paper how to retain the original color of the labels when using them in transformations.

We'll be using a looping macro that fetches data from multiple screens. It may be desirable to extract the color of one particular region in some of the screens of the macro. We'll be extracting the region using an indexed global variable that will get populated automatically as the macro plays. We'll edit the macro source file to extract the color of the particular region in all the screens of interest.

A Custom Java class needs to be developed that will return the color of each extracted label in the macro to a global variable. This class will extend the **AbstractAdvancedCustomScreenRecoListener** class.

The following snapshot displays the **ExtractColors** class that extends the **AbstractAdvancedCustomScreenRecoListener** class:

```
public class ExtractColors extends AbstractAdvancedCustomScreenRecoListener {

    public boolean isRecognized(String settings, IBusinessLogicInformation bl, ECLPS ps, ECLScreenDesc sd) {
        //check to make sure we have a Presentation Space

        if(null != ps){
            IGlobalVariable gv = bl.getGlobalVariable("colors");
            if (null == gv) {
                gv = new GlobalVariable("colors");
            }
```

The above code declares a new global variable named **colors**.

The code snippet below gets the color of the extracted plane and indexes the color of each element in the global variable:

```
char[] colorBuff = new char[2];

ps.GetScreen(colorBuff, 2, i, startCol, 1, ECLConstants.COLOR_PLANE);
int val = colorBuff[0];
System.out.println("Color Val "+val);
gv.add(Integer.toHexString(val));
```

After we are finished with the recording of the macro, we view the source of the macro and look for the first screen tag (**<screen>**) that contains the extracted labels; the colors of which need to be found. For example, consider the screenshot below:

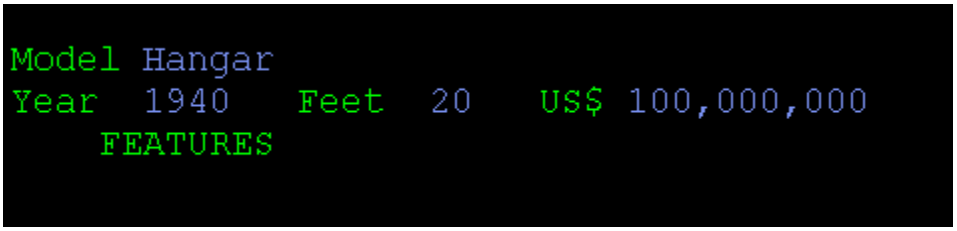
```
<screen entryscreen="false" exitscreen="true" name="ExitMacro" transient="false">
    <comment>
        !visualinfo:x=152;showActions=1;associatedScreenName=ExtractLoanData\Screen1.hsc;y=245;!
    </comment>
    <description uselogic="1 and (!2 and 3 and 4)">
        <oia invertmatch="false" optional="false" status="NOTINHIBITED"/>
        <string casesense="false" col="79" invertmatch="true" optional="false" row="20" value="+" wrap=
        <string casesense="false" col="2" invertmatch="false" optional="false" row="1" value="FCOLR700
        <customreco id="com.ibm.CustomScreenReco.ExtractColors" invertmatch="false" optional="false"/>
    </description>
```

We have added the highlighted line in the above screenshot. We have added the **customreco** tag. The **id** attribute in this tag points to the **ExtractColors** class that was discussed earlier.

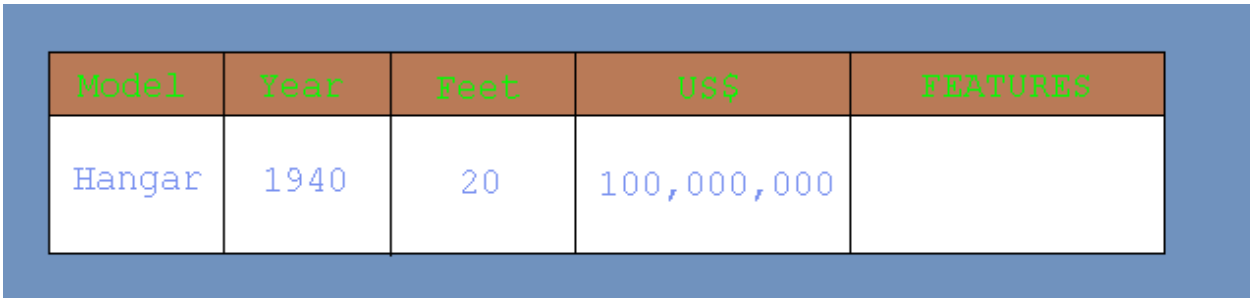
Whenever the macro plays, it will extract the specified regions of the looping screens in the indexed global variable which is defined in the **action** tag of the above screen (not shown). The macro will also extract the color labels of the extracted labels from the screen into the indexed global variable named **colors** because we have used the **customreco** tag in the above screen.

Let's assume that we extracted the labels in an indexed global variable named **values**. Recall that our **ExtractColors** class will extract the corresponding colors in the **colors** indexed global variable. Now if we need to display the extracted labels with their proper colors on a JSP, we fetch both the indexed global variables on the page and write the appropriate logic to assign each label from the **values** global variable its correct color from the **colors** global variable.

For example, our host screen may look like the one below:



By using the above procedure we are able to render the above fields in the same color on the webpage as the host:



Model	Year	Feet	US$	FEATURES
Hangar	1940	20	100,000,000	

Sources:

HATS Hotspot : http://www-949.ibm.com/software/rational/cafe/community/hats

IBM: www.ibm.com

IBM Global Services
Route 100
Somers, NY 10589
U.S.A.
Produced in the United States of America
08-10