Offline Help Files

6.03 Wsprint APIs

In Offline, printing functions are performed by means of a dynamic library file (DLL) called 'wsprint.dll'. This file extends the webscript.dll file and provides facilities to access and use printers through Javascript. For the printing functionality to work correctly, the wsprint.dll file must be loaded in the offline application. Loading the wsprint.dll is done by using the Load command as shown in the following example:

```
load(wsprint);
```

However, it is also necessary to load the wsprintjs file. This file contains the system APIs used by the print function. To load this, the following code should be used:

```
System.LoadFile('wsprint.js');
```

Please note that the Wsprint extension is only supported with Offline version 3.1.40.0 and above. Earlier versions do not support this extension. In the following example, we will create a print test using the wsprint extension. The code is as follows:

```
function printTest( )
{
var a, b;
try { a = new Printer( );
var iTOP = 20;
var iBOTTOM = 20;
var iLEFT = 20;
var iRIGHT = 20;
var iColor = 130;
var fColor = 240;
var hColor = 90;
var arrColor = [iColor,fColor,hColor];
var iMargin = [iTOP,iBOTTOM,iLEFT,iRIGHT];
var aFont={type:"Verdana",size:12,underline:true,color:arrColor};
var bFont={type:"Verdana",size:10,underline:false,color:"Blue"};
a.print({font:aFont,margin:pMargin,spacing:20});
a.printCenterAlign(xFld1 + "\n");
a.print({font:bFont,margin:pMargin}, xFld2 + "\n" + xFld2 + "\n" + xFl
d3 + "\n" + xFld4 + "\n" + xFld5 "\n" + xFld6 + "\n" + xFld7 + "\n" +
xFld8);
a.printRightAlign(xFld6 + "\n" + xFld7 + "\n" + xFld8);
a.print({cmd:'FLUSH'});
 }
catch(err) { System.TraceOutput(err);

} }
```

In the preceding example, the first thing we did was to construct a new print object using the New Printer function, as follows:

```
a = new Printer( );
```

If the new printer is a network printer, the full path name must be specifed as shown below:

```
a = new Printer('\\\\MURMILLO\\HP LaserJet 2300dtn PCL6');
```

Four backslashes as in this example are used to specify the network address since '\' is an escape character in Javascript. If the printer is not a network printer, the following syntax is used:

```
a = new Printer('Microsoft XPS Document Writer');
```

Since this printer is not a network printer, only the name needs to be specifed. If the printer is the default printer, the below syntax can be used as in our code example:

```
a = new Printer( );
```

Once the printer object is created, it is necessary to define the position variables. These will be used later for the margins. The code is as follows:

```
var iTOP = 20;
var iBOTTOM = 20;
var iLEFT = 20;
var iRIGHT = 20;
```

When these are used to create a margin, the code is as follows:

```
var iMargin = [iTOP, iBOTTOM, iLEFT, iRIGHT];
```

Once the position variables are defined, it is necessary to define the font variables.

```
var iColor = 130; var fColor = 240; var hColor = 90;
```

Font objects can also be directly created and values assigned as for aFont and bFont below:

```
var aFont={type:"Verdana", size:12, underline:true, color:arrColor};
var bFont={type:"Verdana", size:10, underline:false, color:"Blue"};
```

Once these objects are created and the values assigned, it is necessary to pass the font and margin objects along with the desired text to the print object. The code for this is shown below:

```
a.print({font:aFont, margin:pMargin, spacing:20});
a.printCenterAlign(xFld1 + "\n");
a.print({font:bFont, margin:pMargin}, xFld2 + "\n" + xFld2 + "\n" + xF
ld3 + "\n" + xFld4 + "\n" + xFld5 "\n" + xFld6 + "\n" + xFld7 + "\n" +
 xFld8);
a.printRightAlign(xFld6 + "\n" + xFld7 + "\n" + xFld8);
```

We then need to send the print object to the printer, which is accomplished with the following code:

```
a.print({cmd:'FLUSH'});
```

Finally, we need to include some error handling to catch any errors and inform the user. This code is in two parts - one to catch the error and one to display the error in the debug window. This second part uses the System object and the TraceOutput method. The code is as follows:

```
catch(err) { System.TraceOutput(err); }
```

Unique solution ID: #2077
Author: sarvani.kusuri@guixt.com
Last update: 2021-06-03 18:26