Development-CSS

Overview

CSS stands for Cascading Style Sheets. It's provides a mechanism to separate the presentation of your HTML page from the logical structure and content of your HTML page.

Referencing a Style Sheet

Style Block

The easiest way to include CSS in your WebObject application is put a style block in your <head> tag:

```
<style type="text/css">
  table {
    border-style: solid;
    border-color: black;
    border-width: 1px;
  }
</style>
```

Static Reference

Another easy way to use CSS in your WebObjects application is to include a static reference to it with:

```
<link rel = "stylesheet" type = "text/css" href = "/webserver/relative/path/to/mystyles.css">
```

However, if you're a WO developer, you know static resource references feel dirty. Don't worry, there are alternatives.

Project WOnder

Project WOnder provides ERXStyleSheet, which is a stateless component with a template that can generate link tags for you.

Mike Schrag's MDTStyleSheet

The following dynamic element can be used to include a stylesheet in your page. It supports the bindings "rel", "type", "href", "src", etc similar to a WOImage.

```
import com.webobjects.appserver.WOElement;
import com.webobjects.appserver._private.WOConstantValueAssociation;
import com.webobjects.appserver._private.WOHTMLURLValuedElement;
import com.webobjects.foundation.NSDictionary;

public class MDTStyleSheet extends WOHTMLURLValuedElement {
  public MDTStyleSheet(String _name, NSDictionary _assocationsDictionary, WOElement _template) {
    super("link", _assocationsDictionary, _template);
    if (_associations.objectForKey("rel") == null) {
      _associations.setObjectForKey(new WOConstantValueAssociation("stylesheet"), "rel");
    }
    if (_associations.objectForKey("type") == null) {
      _associations.setObjectForKey(new WOConstantValueAssociation("text/css"), "type");
    }
  }

  protected String urlAttributeName() {
    return "href";
  }
}
```

Stylesheet Component

Here is a trivial example on how to define style sheets properties at runtime:

Simply create a component with your style sheets properties:

```
<style type = "text/css">
  a { color: <webobject name = "LinkColor"></webobject>; text-decoration:none; }
  a:hover { color: #ff9933; }
  a:visited { color: <webobject name = "LinkColor"></webobject>; }
  a:visited:hover { color: #ff9933; }

  body { margin-top:0px; margin-bottom:0px; margin-left:0px; margin-right:0px; background-color:#FFFFFF; text-
align:left; }

  input { font: 100% Verdana, Arial, san-serif; }

  .Title { font: <webobject name = "TextFont"></webobject>; }
  .Label { font: <webobject name = "LabelFont"></webobject>; }
  .WhiteLabel { font: <webobject name = "LabelFont"></webobject>; color:#666666; }
  .Text { font: <webobject name = "TextFont"></webobject>; }
  .MonoText { font: <webobject name = "MessageFont"></webobject>; }
  .Quote { font: <webobject name = "MessageFont"></webobject>; font-style: italic; margin-left: 20px; }

  .Line { height:1px; background-image:url(<webobject name = "LineUrl"></webobject>); }
  .Space { height:8px; }

  .Highlight { background-color:#cccccc; }
  .DarkHeader { background-image:url(<webobject name = "DarkHeaderUrl"></webobject>); }
  .Header { background-image:url(<webobject name = "HeaderUrl"></webobject>); }

  .Margin { width: 40px; vertical-align:top; }
  .Full { width: 100%; height: 100%; text-align:left; vertical-align:top; }
  .FullWidth { width: 100%; text-align:left; vertical-align:top; }
  .FillerWidth { width: 99%; text-align:left; vertical-align:top; }
  .FillerHeight { height: 99%; }
  .HalfWidth { width: 49%; text-align:left; vertical-align:top; }
  .OneThirdWidth { width: 33%; text-align:left; vertical-align:top; }
  .TwoThirdWidth { width: 66%; text-align:left; vertical-align:top; }
  .FixColumn { width: 250px; text-align:left; vertical-align:top; }

  .AltMargin { width: 40px; text-align:left; vertical-align:top; background-image:url(<webobject name = "AltUrl"
></webobject>);
       background-position: center center; background-repeat: no-repeat; }
</style>
```

And use some wod for the parameters:

```
LinkColor: WOString{
  value = linkColor;
};

LineUrl: WOString{
  value = lineUrl;
};

DarkHeaderUrl: WOString{
  value = darkHeaderUrl;
};

HeaderUrl: WOString{
  value = headerUrl;
};

AltUrl: WOString{
  value = altUrl;
};

LabelFont: WOString{
  value = labelFont;
};

TextFont: WOString{
  value = textFont;
};

MessageFont: WOString{
  value = messageFont;
};
```

Related Links

- CSS Programming
- CSS on Wikipedia
- W3 Schools on CSS