Raster Image Overview

Matt Ernst
Pacific University

Follow this and additional works at: http://commons.pacificu.edu/inter02

Recommended Citation

This Article is brought to you for free and open access by the Interface: The Journal of Education, Community and Values at CommonKnowledge. It has been accepted for inclusion in Volume 2 (2002) by an authorized administrator of CommonKnowledge. For more information, please contact CommonKnowledge@pacificu.edu.
Raster Image Overview

Rights
Terms of use for work posted in CommonKnowledge.

This article is available at CommonKnowledge: http://commons.pacificu.edu/inter02/69
Separation of Content & Design Part I: Transitioning To XHTML & CSS

Posted on November 1, 2002 by Editor

By Ken Westin <webmaster@pacificu.edu>

Author Bio

INDEX:

.01 The Drama
.02 CSS/XHTML To The Rescue
.03 Let’s Put The “X” In XHTML
.04 Styling With CSS

.01 The Drama (return to index)

You finally finished the 100 page website for your boss. You organized the content yourself, did all of the graphics and even marked up all the pages. Now the moment of truth, you present it to your boss …

For the site you used a white background and blue and green graphics. To control all of the fonts you used <font> tags using Times as your font and the size set to “2″. Then it happens, you finally show your hard work only to discover that the marketing folks are coming out with a new marketing campaign with all new pamphlets, logo and color scheme and the website must match it. To top it all off your boss says the text is too small and he heard that Verdana is easier to read on the web than Times. And of course your boss opens it in Netscape 6 on a Macintosh, when you wrote the page for IE 5.5+ on Windows, so your layout is completely blown apart. Through your tears you try to explain to them that you were up until 4:00 am finishing the site and really don’t want to do it again… your cries fall upon deaf ears.

The case above is what happened in the late 90’s (and still today) with the big rush to the web.
The “Browser Wars” were in full effect as competing browsers interpreted HTML differently so developers had to develop multiple versions of the same site which took a great deal of manpower and money. It was due to these browser incompatibilities that Web Standards came into being; it was an attempt to get the browser makers to play nice. It is through the efforts made by the W3C and Web Standards Project (WaSP) that a super mark-up language called X-HTML was born. X-HTML is similar to its big brother XML, in fact it is XML, but it is still has the same functionality of the HTML you know and love. The benefit of the X-HTML is that it unlike HTML it has a standard way that it is to be rendered by web browsers.

With old HTML you design your layouts and then hope and pray that it will work in all of the browsers. Coders would rely on old “hacks” to trick HTML into designing their pages using complex nested tables, spacer gifs, and background images to name a few. HTML was never meant to be used as a design tool, HTML is a structural language. It is a bit like asking the contractor who built the foundation of you house to do your “Feng Shui”, he might get the job done, but it may not look quite as elegant as you had hoped and may use unorthodox methods.

But now it is almost 2003 and we all know better now right... ? Wrong. Here we are 2 years into the new millennium and we are still developing websites like we did back then, only now we don’t have as good of an excuse. True if you are still targeting the older browsers such as Netscape 4.7, as we are doing here at Pacific University you will still need to rely on table based layouts, however they do not need to be as complex as they were in the past as the majority of the design of your pages can be done in Cascading Style Sheets.

CSS/XHTML To The Rescue (return to index)

Using X-HTML and CSS to design web pages is much simpler in the long run, particularly for larger sites as you control the look and feel of your pages with a single text file. Using X-HTML will pretty much guarantee that the page you design will look similar in all standards compliant browsers including IE 5.5+, Mozilla 1.x, Netscape 6-7, Opera 6 and more to come in the future. CSS will allow you to add some style to your pages, reduce the size of your pages and save you some sanity as well.

Let’s Put The “X” in XHTML (return to index)

Well the good news is that if you know HTML you know X-HTML as it is very similar. The only differences are that X-HTML follows the rules of XML. XML stands for eXtensible Markup Language, not quite sure why it is not called EML, but I guess XML is a bit sexier. XML is a language that is used by programmers and developers for various purposes, but basically it is just a simple text file with data and some rules applied to it. I will not go great detail about XML at this point as it is beyond the point of this discussion, however it will be important to realize that X-HTML is an implementation of XML.
The first thing we need to do for our XHTML page is make a document declaration:

```html
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
```

Put this on the very top of your page; this states that you are going to be using XHTML 1.0 Transitional as put forward by the W3C. You don’t need to memorize it, you can just copy and paste it into the very top of your document.

Next you declare your XML namespace in your beginning HTML tag like this:

```html
<html xmlns="http://www.w3.org/1999/xhtml">
```

Since there are many implementations of XML, we need to tell the browser which one we are using. The namespace is declared by using the URL of where the specification lives. If you go to http://www.w3.org/1999/xhtml you will be taken to a webpage that serves as the specification for XHTML, the browser does not link to this page or access it anyway, the folks at the W3C just figured that it would be best to name the namespace with the URL of where the specification lives.

We said the XHTML is like HTML, but with rules applied. These rules come from XML, it says that every tag that starts must have an ending tag. So for example here is the old way people would use the paragraph tag `<p>`:

```html
<p> I love XHTML and CSS, it makes my life easier and allows me to spend more time with my cat. </p>
```

As you can see we can tell where the paragraph starts, but we are not telling the browser were the paragraph ends. However, more browsers will “figure” this out for you and will assume that since you are starting a new paragraph the old one must end. This has become what is known as “quirks” mode, where you are giving control over to the browser to dictate how best to render your pages. Now, with XHTML we don’t let the browser tell our paragraphs what to do, we tell the browser exactly where the paragraphs end like this...

```html
<p> I love XHTML and CSS, it makes my life easier and allows me to spend more time with my cat. </p>
```

Now we are writing XHTML. What about tags that we know don’t have an ending like the break tag `<br>`? Well in XML syntax this is done by putting a forward slash after the “br” like this `<br/>`, this tells the browser “hey this is where I start and end, you don’t have to look for my ending tag”. Now there is a small catch with this, since older browsers don’t know what “<br/>” means we have to trick them. If we put a space between the “br” and “/” older browsers will think it is the old `<br>` tag it understands, so we write it like this “<br />” and viola you have an XHTML tag.
that old browsers think is good old fashioned HTML. Another tag that is like this is the image tag `<img>`. Here is the old way:

```html
<img src="my_fat_cat.jpg" width="250" height="100"/>
```

Now the XHTML version:

```html
<img src="my_fat_cat.jpg" width="250" height="100" alt="My Cat Nabi"/>
```

As you can see there are a few differences, but the syntax is similar to what we are used to in HTML (if you are used to HTML of course). The differences as you will notice is the use of the “alt” attribute, in HTML is was considered good style to use these alt tags, however in XHTML this is mandatory. There are a few other differences between XHTML and HTML, but this should get you started for now.

.04 Styling With CSS (return to index)

Now we get to the fun stuff. Remember the `<Font>` tag you have grown to know and love? Well it is time to bury it and never use again. I will show you a better way to control your fonts using CSS (Cascading Style Sheets). If you are coding for browsers below 4.0, CSS will not work so well as these browsers were made before the CSS standards were implemented in most browsers. If you are coding for Netscape 4 you will find that it supports most CSS, but not all as it was implemented poorly by Netscape. However, the stuff you will see here will work in Netscape 4 without a problem.

Let’s start with an easy one. Let’s say we want all of the paragraphs on your page to have red text and be sized to number “2″. Then your code for each paragraph will look something like this:

```html
<p><font color="red" size="2" face="Verdana, Arial, Helvetica, sans-serif">
This is some text in a paragraph that I want to style
</font></p>
```

Not too difficult really, but the fact that you have to do this for every paragraph can be a bit of a pain, especially if you have to keep changing it around. Instead let’s try this with CSS. In the head of your HTML Document type or copy and past this:

```html
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<style type="text/css">

p { font-family: Verdana, Arial, Helvetica, sans-serif; font-size: 13px; color: #F00; }
```

http://bcis.pacificu.edu/interface/?p=2643
Now for the explanation. First we have our document type definition up on time stating that our document is in XHTML 1.0 Transitional. We then have our html tag with our XML Namespace as “http://www.W3.org/1999/xhtml”. Then in our <head> tag we add a new tag we have not seen before, this is our style tag. In this tag we tell the browser what language we are using for the style in the "type" attribute, so we put “text/css”.

Now the good stuff. I put an HTML comment block at the start and end of my CSS code because if someone with a really old browser hits this page, the browser will render this in the web page, so we hide it from these old browser by putting the comments in place. Next, we tell the browser what tag we are defining and since we just want to style the <p> tag right now we will put “p”. The stuff that follows this is the style definition. We start the style definition with a curly brace “{" and end it with a curly brace “}”, remember that with XHTML and CSS everything that starts must end. We then define the fonts we would like to use for the <p> tag. First I want Verdana and if the user doesn’t have it let’s use Arial, or Helvetica and if they don’t have any of those (probably Linux users) use whatever sans-serif font you can find. Next I say I want the font size to be 13 pixels in size and the color to be #F00 which is shorthand CSS for red. Now, everywhere that we put the <p> tag the text will conform to these rules.

It may seem a little complex now, but once you get under the hood you will realize that CSS will save you lots of time and heartache. Lets say you finish your page and decide you would rather use the Times font instead and you want it to be 16 pixels in size and the text to be black, just change the CSS definition to this:

\[
p \{ \text{font-family: Times, serif; font-size: 16px; color: #000;} \}
\]

Now we are telling the browser to make all of the text in the <p> tag Times and if they don’t have it just pick a serif font on the system and make it 16 pixels in size and black (#000). You just saved yourself a good half hour of work for this page. Now you can also create a style sheet that controls the entire look of a site. This is done by linking and external style sheet to your page. Here is how you link the external style sheet.

\[
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<link rel="stylesheet" href="mystyle.css " type="text/css" />
\]
So if you just copy the CSS that you had in the body of your document, open up a text file and paste it in like so minus the style declaration:

```css
p { font-family: Times, serif; font-size: 16px; color: #000; }
```

Then save this in the same directory as your pages and make sure that this line of code is in the head of every page:

```html
<link rel="stylesheet" href="mystyle.css " type="text/css" />
```

All of your pages will then have the same styles. An example of this is the News & Events section of the Pacific University website [http://www.pacificu.edu/news_events/](http://www.pacificu.edu/news_events/), go ahead view the source code of this page. You will see that the page is calling a stylesheet located at:

[http://www.pacificu.edu/Global/css/master.css](http://www.pacificu.edu/Global/css/master.css)

Go ahead and take a look at this stylesheet, your browser will open up the stylesheet as a text file, it is quite a bit more complex than what we have done so far, but here you can see how CSS is being used to control the look and feel of several pages and eventually an entire site. The fact that the browser only downloads the CSS sheet once as it is cached in your browser allows you to chop down the size of your pages as well as spicing them up a bit. You may notice as well that the folder tabs are not graphics, but plain text controlled by CSS—the entire navigation bar is under 15K, including the logo. I will go over how to make the mouse over effects with CSS later, but for now practice with the example I have above and pick up a book on CSS, you will be glad you did. When you use XHTML in combination with CSS, you will save yourself a great deal of time and headaches.

This entry was posted in Uncategorized by Editor. Bookmark the permalink.

7 THOUGHTS ON “SEPARATION OF CONTENT & DESIGN PART I: TRANSITIONING TO XHTML & CSS”

naija social network on January 30, 2014 at 11:37 AM said:

It's like you read my mind! You look to know much about this, like you wrote the book in it or something. I think that you can do with some pics to drive the message property a bit, but rather than that, this is great blog. A good read. I am going to definitely be back.
เลือกผู้แพร่รู้
on January 31, 2014 at 8:41 PM said:

I really like it when fools come together and share views.

Great site, continue the good work!

plotka
on February 1, 2014 at 1:46 AM said:

I will immediately clutch your rss as I cannot in finding your email subscription link or newsletter service.

Kraków
on February 1, 2014 at 2:17 AM said:

your attendees and take pictures.

naija
on February 3, 2014 at 1:37 AM said:

I also agree with you. i believe that there are lots of lessons to be learned from this book. By not reading the book, we miss out on some elements that are a certain impact to our life. However, I do believe you ought to be a certain maturity so that you can get from this book what you need

temat
on February 3, 2014 at 1:56 AM said:

I am often browsing on-line for posts that will benefit me. Thx!
Nigeria social network on February 5, 2014 at 12:17 AM said:

Thanks significantly for providing men and women with this sort of a remarkable opportunity to read important reviews from here. It’s always so superb plus packed with a great deal of fun for me personally and my office peers to visit your internet site minimum Three times every week to find out the new elements you have. And indeed, I’m also truly satisfied as part of your astonishing tips served by you. Some Two data in this posting are totally the simplest we’ve ever had.