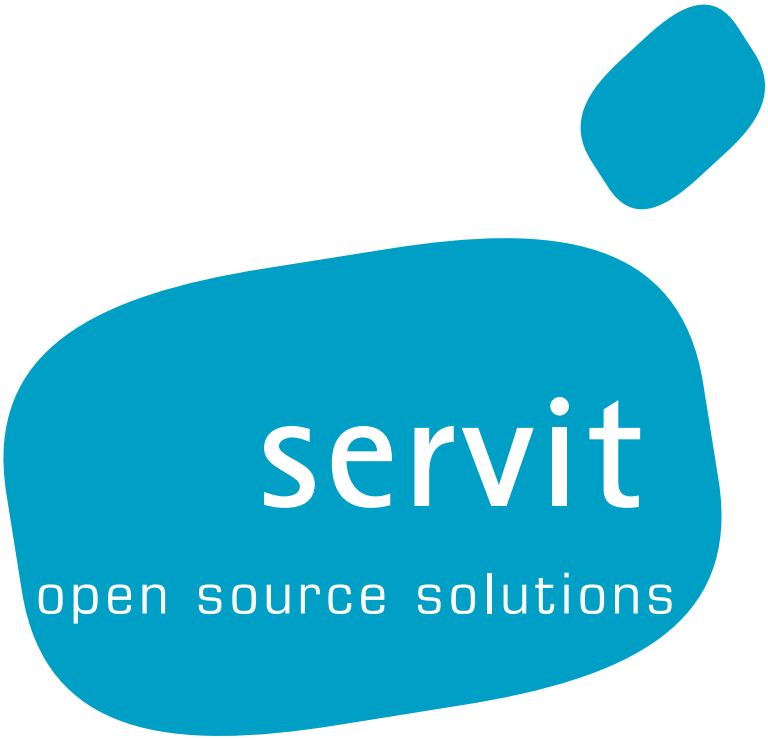

Servit Tech Newsletter 002

Layout & Interaction for Websites with CSS & Javascript

Created by Servit Open Source Solutions on 3.6.2008

Table of Contents

1. Introduction

After covering the basic principles of the World Wide Web and HTML in the last issue of Servit Tech Newsletter we will have a look this time at two technologies which make all the fancy designs and the fast and interactive websites possible: Cascading Style Sheets and Javascript. Enjoy learning and dive deeper into the mysteries of the web.

2. Layout with Cascading Style Sheets (CSS)

2.1. Why CSS?

We saw in the previous issue of Tech Newsletter that with HTML we can build web pages, at least the content and its logical structure. In order to give them a layout and design you can use Cascading Style Sheets.

If you have played around with HTML you will have noticed that it offers means not only to create content and structure but to style a web page as well. There are attributes for colors, backgrounds, etc. And you can use invisible tables to position content on a page. So you may ask, what for do I need CSS?

There are three shortcomings when using HTML for styling and layout:

- HTML offers very limited styling capacities
- With HTML only you are mixing code for style and code for structure and content in the same document, yes even on the same lines. There are some pretty good reasons for keeping code for layout and code for structure and content completely separate.
- When positioning content using invisible tables you break up the (logical) document structure. You're abusing logical elements (table, list, etc) for design purposes. So tables are not used to display tabular data when using invisible tables for positioning pictures properly aligned on the page something tabular.

While the last point is kind of philosophical and can be subject of discussions it is nowadays seen as bad style to use HTML elements for styling purposes.

2.2. How it works

CSS is basically code describing the style of a web page. By writing CSS code you can tell the browser how to display your HTML page. You can for example define a border or background for a certain element, or you can position an element on any part of the page.

CSS code can be put in three different places:

- inside HTLM tags (as an attribute called “style”)

- in a separate CSS code block in the HTML file
- in a separate CSS file

In most cases it's better to put the CSS code into a different file than the HTML code. That way a true separation of content/structure and style is achieved. Furthermore having a separate CSS file permits to have the same style to be applied to all web pages of a web site and to have one central place to edit it.

In this article we will only cover working with CSS using separate CSS files. The other methods differ only slightly from it. Please check the references at the end of the document for more information them.

2.3. CSS code

CSS code also called CSS syntax consists of rule-sets each describing the style of certain elements. Each rule-set, consists of one or more selectors followed by a declaration block. The selectors define to which HTML elements the styles defined in the declaration block apply. This way it is for example possible to define a certain style and apply it to all links on a website.

```
a{
  color: black;
  font-size: 10px;
  font-family: arial;
}
```

Example code to style all links (HTML: <**a** href="www.site.com">site</**a**>).

With the selectors it's possible to target either certain element types like links, tables, etc., or elements with specific classes or IDs. When defining an element in HTML it's possible to define a unique ID for it or to assign it a class. With the unique IDs it's possible to target a specific elements on a page. With classes however you can apply a style to all element with a certain class assigned.

```
p{
  border:1px solid black;
  background-color:red;
}
```

*CSS rule applies to all paragraph **elements**, for example:*

```
<p>I'm a red paragraph wit black border</p>
<p>Me too!</p>
<div>Me not, the rule doesn't apply for me. :( </div>

-------------------------------------------------------------------
#elem123{
  font-size:100px;
}
```

*CSS rule applies to the element with **ID** "elem123":*

```
<div id="elem123">I'm really big!</div>

-------------------------------------------------------------------
.orangeBorder{
  border:1px solid orange;
```

```
}
```

*CSS rule applies to all element with the **class** "orangeBorder":*

```
<div class="orangeBorder"> I have an orange border. </div>
<p> Me not. :( But the following image has one:</p>
<img class="orangeBorder" src="photo.jpg">
```

Examples of CSS rules targeting HTML elements by element type, ID resp. class.

2.4. Exercise

Here comes a little exercise to get familiar with CSS and recapitulate HTML.

Create a simple HTML page with at least a table. Afterwards create a CSS file and style the content of the HTML page. Don't forget to put a reference to the CSS file into the HTML file. That's how you tell the browser to apply the styles from that file.

- ✔ Create an HTML file
- ✔ Put at least following table into the new web page:

```
<table >
   <tr>
     <th>Country</th> <th>Dialing Code </th> <th>Currency</th>
   </tr>
   <tr>
     <td>Switzerland</td><td>+41</td><td>CHF</td>
   </tr>
   <tr>
     <td>Germany</td><td>+49</td><td>EUR</td>
   </tr>
   <tr>
     <td>Jordan</td><td>+962</td><td>JOD</td>
   </tr>
 </table>
```

- ✔ Create a CSS file
- ✔ Put a reference to the CSS file into the HTML file
  (<link rel="stylesheet" type="text/css" href="styles.css">)
- ✔ Create a CSS rule for all tables in the CSS file (table{background-color: red;})
- ✔ Assign a class to some table cells (<td class="red">...</td>)
- ✔ Create a CSS rule for that class (. red{color:red})
- ✔ Create more styles and play around with the styling of the elements.

If you need help and examples for above task, check also following resources:

2.4.1. Creating HTML files

See the last newsletter issue and get examples at:
http://de.selfhtml.org/html/allgemein/grundgeruest.htm (de)
http://www.htmldog.com/guides/htmlbeginner/conclusion/ (en)

2.4.2. Tables

http://de.selfhtml.org/html/tabellen/aufbau.htm (de)
http://www.html.net/tutorials/html/lesson10.asp (en)

2.4.3. Creating CSS files

Create a text file (same as you do for an HTML file), add the extension ".css" to the file name and write your CSS rules into the file. Don't forget to reference the CSS file

in the HTML file:
http://de.selfhtml.org/css/formate/einbinden.htm#separat (de)
http://www.htmldog.com/guides/cssbeginner/applyingcss/ (en)

2.4.4. CSS rule for tables

http://de.selfhtml.org/css/eigenschaften/tabellen.htm (de)
http://de.selfhtml.org/navigation/css.htm#tabellen (de)
http://w3schools.com/css/css_table.asp (en)
http://w3schools.com/css/css_reference.asp#table (en)

2.4.5. CSS References

http://de.selfhtml.org/navigation/css.htm#tabellen (de)
http://w3schools.com/css/css_reference.asp (en)

2.5. Summary

Cascading Style Sheets (CSS) provide a mechanism to style web pages (HTML pages). CSS is a style sheet language describing the presentation of an HTML document.

The advantages over styling trough HTML attributes are:

- extensible styling possible
- separation of content/structure and styling (presentation)
- possibility to have a central stylesheet with rules for all pages of a web site.

This makes maintenance and changes in the design of a web site easy if CSS is used centrally. By exchanging or editing the stylesheet file the changes are immediately applied to all pages.

Zen Garden makes use of this. The web site displays always the same HTML page but with different stylesheets. It just changes the reference to the CSS file. Experience it yourself:
http://www.csszengarden.com (en)
http://www.csszengarden.com/tr/deutsch/ (de)

2.6. References

– CSS @ SelfHTML (de)
http://de.selfhtml.org/css/index.htm

– CSS Reference @ SelfHTML (de)
http://de.selfhtml.org/navigation/css.htm

– CSS @ Wikipedia (de)
http://de.wikipedia.org/wiki/Cascading_Style_Sheets

- CSS @ w3schools (en)
  http://w3schools.com/css/default.asp
- CSS Reference @ w3schools (en)
  http://w3schools.com/css/css_reference.asp
- CSS @ Wikipedia (en)
  http://en.wikipedia.org/wiki/Cascading_Style_Sheets

3. Interaction with Javascript (JS)

3.1. User – Server communication in the beginning of WWW

In the old days all websites were more or less static. If there was some interaction between the user and the site it was necessary to reload the page. For example when filling out a form on a page it was necessary to hit the send button. The filled in data was then sent from the users browser to the web server and processed there. If the submitted data was incorrect (e.g. invalid email address), an error message was shown on the newly loaded page.

The browsers where not doing more than displaying pages they received from web servers and sending back form data the user had filled in. All “intelligence” was on the server and the interaction between user and server based upon loading and reloading web pages. This takes time and is annoying especially when it is only about checking some data in a form.

3.2. Javascript — bringing intelligence to browsers

With the introduction of Javascript (JS) intelligence was brought closer to the user. It became possible to run certain processes and tasks in the browser itself without reloading the web page. The browsers got smarter.

With JS it is for example possible to check the validity of an email address while the user is typing it into the form. No need to send it to the sender only to validate the data. Of course this is not the only application. Javascript has a lot more to offer. It can take input from the user, edit the displayed web page, make calculations and more. All this is done within the browser. No need to send data back and forth to the server for this tasks.

3.3. How it works

Javascript is a programming language. It is not a compiled language but a script language. This means the instructions are not translated (compiled) into series of 1 and 0. The instructions written by the programmer stay as they are and are sent to the computer which interprets (reads) and executes them.

This means that a Javascript program consists of a text file which is sent to the

browser. This file contains the instructions for the browser which it interprets and does what it is told. One consequence of this is that JavaScript code can't be kept secret. If a browser can download and read the instructions, humans can do as well.

```
document.onkeydown = ButtonPushed;

function ButtonPushed (e) {
  alert("Stop pushing me!!");
}
```

Javascript code example

Above code is an example for a small JavaScript application. It is downloaded by the browser from the web server along with the web page. The code tells the browser to display a message to the user (`alert(...)`) if he pushes a button while the web page is displayed.

3.4. Problems with Javascript

Javascript brings not only improvement but also new problems. One concern is security. Before Javascript the World Wide Web was only about displaying web pages. Now the web servers send code to the browser which runs on the users computer. It's not anymore only the user who takes action but Javascript programs who are downloaded along with web pages and executed mostly without the user knowing. The owner of a website can send his visitors malicious code which is executed by the browser. Luckily there are several limitations about what Javascript is allowed to do on the users computer. It has for example no access to the files on the computer nor other system resources outside the browser.

In the early days of Javascript those restrictions were few and the security holes in the browser numerous. But in the meantime Javascript matured and can be considered as safe. Still, caution is advised when visiting dubious sites.

Another problem with Javascript are the inconsistencies on the different browsers. Javascript is an interpreted language. This means the instructions are written as text and interpreted by the browser. Especially in the beginning of the Javascript era every browser had his own way of interpreting those instructions. Some instructions where only working on Netscape but unknown by Internet Explorer and vice versa. This lead to a lot of headaches for web developer. Luckily this situation has improved with maturation of browsers.

Because of the security concerns users are able to deactivate JavaScript in their browser. Furthermore there are some browsers which don't support JavaScript. Although both groups are a minority it is still important to keep in mind when developing web sites, that not all users are able to execute JavaScript and that all content has therefore to be accessible without JavaScript.

3.5. Web 2.0, Ajax — Javascript changed the WWW

Web 2.0, a term which became popular in the last years, describes the new way of

using the World Wide Web. Usability, user friendliness, creativity, information sharing and collaboration are just some of the terms associated with Web 2.0. Much of the Web's innovation and evolution during the Web 2.0 era has relied upon JavaScript, especially Ajax.

That's why Javascript became very popular in the last years. It allows much faster and smoother interaction with a user. With Ajax it became even unnecessary to reload a web page at all.

Ajax is a technique to load content from the web server in the background using Javascript. The user doesn't have to wait anymore for a new page to be loaded. He can continue working and Javascript gets the data in the background and updates the page afterwards with the new content.

With Ajax it became possible to create web pages which resemble more to desktop applications (like Word, Excel, etc.) than what web pages used to be in the beginning.

See some examples of what Ajax resp. Javascript can do:
http://www.ajaxdaddy.com/

3.6. Summary

Javascript is a programming language which is executed on the browser. Javascript code is downloaded from the web server along with the web page. The code is human readable and interpreted by the browser.

It brings a fast and smooth user experience by allowing data processing and user interaction with the web site on the users computer. Instead of having all intelligence on the server it is brought closer to the user, to the browser.

Ajax, which is based on Javascript, almost eliminates the need for reloading a web page in order to allow interaction. It brings the user experience with web pages very close to desktop applications.

3.7. References

– Ajax Examples
  http://www.ajaxdaddy.com/

–

– Javascript @ Wikipedia (de)
  http://de.wikipedia.org/wiki/JavaScript

– AJAX @ Wikipedia (de)
  http://de.wikipedia.org/wiki/AJAX

– Javascript @ SelfHTML (de)
  http://de.selfhtml.org/javascript/index.htm

–

- Javascript @ Wikipedia (en)
  http://en.wikipedia.org/wiki/JavaScript
- AJAX @ Wikipedia (en)
  http://en.wikipedia.org/wiki/AJAX
- Javascript @ w3schools (en)
  http://w3schools.com/js/default.asp