# THE POWER OF DYNAMIC HTML

- *Dynamic Content with HTML, CSS and Javascript*

**Written by Apilak Bühl**

**Matr.-Nr.: 851731 (AI-6)**

**University of Applied Sciences**
**Kaiserslautern / Zweibrücken**
**Germany**
**Email: apbuehl@debitel.net**

**Attachment: examples of DHTML**
**Please start with index.html!**

**Or open: http://home.debitel.net/user/apbuehl/dhtml/ to view the examples.**

## *1. Preface*

**This seminar work targets to all developers and users who want to build webpages with dynamic client side content. This paper includes a project with some sample pages, which I will refer to.**

## *2. Overview*

1. **Preface**
2. **Overview**
3. **Introduction**
4. **DHTML-Demo**
5. **Cascading Style Sheets**
6. **Cross-Browser JavaScript**
7. **Sources**

## *3. Introduction*

**DHTML is a entirely client-side technology and gives you the possibility to add more interactivity compared to plain HTML webpages. Most web interfaces at the present, uses only a few features of the wide range supported by the browsers. Although the underlying technologies of DHTML (HTML, CSS, JavaScript) are standardized, the manner in which Netscape and Microsoft have implemented them differ dramatically. For this reason, writing DHTML pages that work in both browsers (referred to as cross-browser DHTML) can be a very complex issue. Programming a own DHTML-API (Application Programmable Interface) can solve the above mentioned problems, so there are no reasons any longer, why not to improve the look and feel of your web interfaces by a 'state of the art technology' like DHTML.**

## *4. DHTML-Demo*

**All of the example pages has been coded by hand from scratch using a simple text editor and a copy of SELFHTML for syntax reference (no WYSIWYG editors were used).**

**The example pages are `only fun pages′ to show some possible ways to create and program dynamic HTML. This paper focuses primarily on the JavaScript issues involved in DHTML. It only covers the portions of CSS-P Cascading Style Sheets Positioning) and JavaScript, that can be used in both Netscape and Internet Explorer.**

**Please use Mircosoft Internet Explorer 4+ to view the DHTML-Demo, because of one reason: it runs faster than Netscape. But nevertheless the example pages are viewable under Netscape 4+ too. You have to enable Javascript and allow popup-windows to view the pages correctly. If you want to start the example pages, simply double-click `index.html′.**

**The following screenshots gives you a first impression of the DHTML sample pages. There are four different example pages:**

**Figure 1: ,INTRO′**

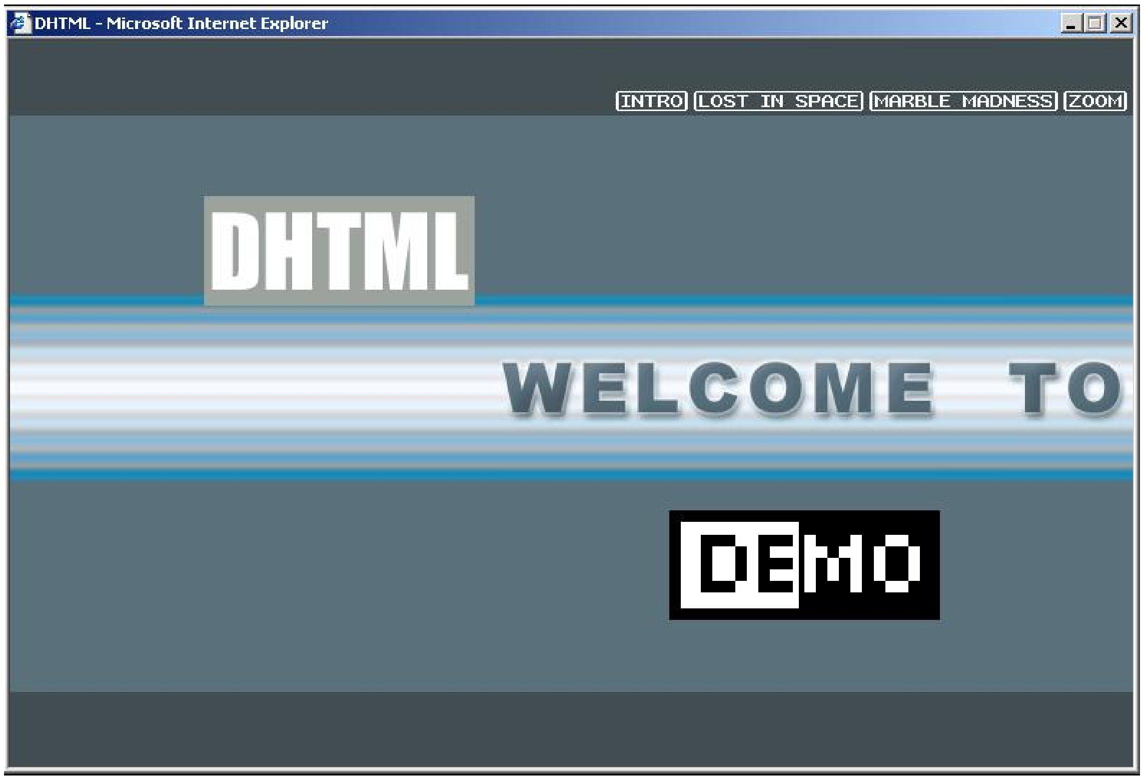


Figure 2: ‚LOST IN SPACE'

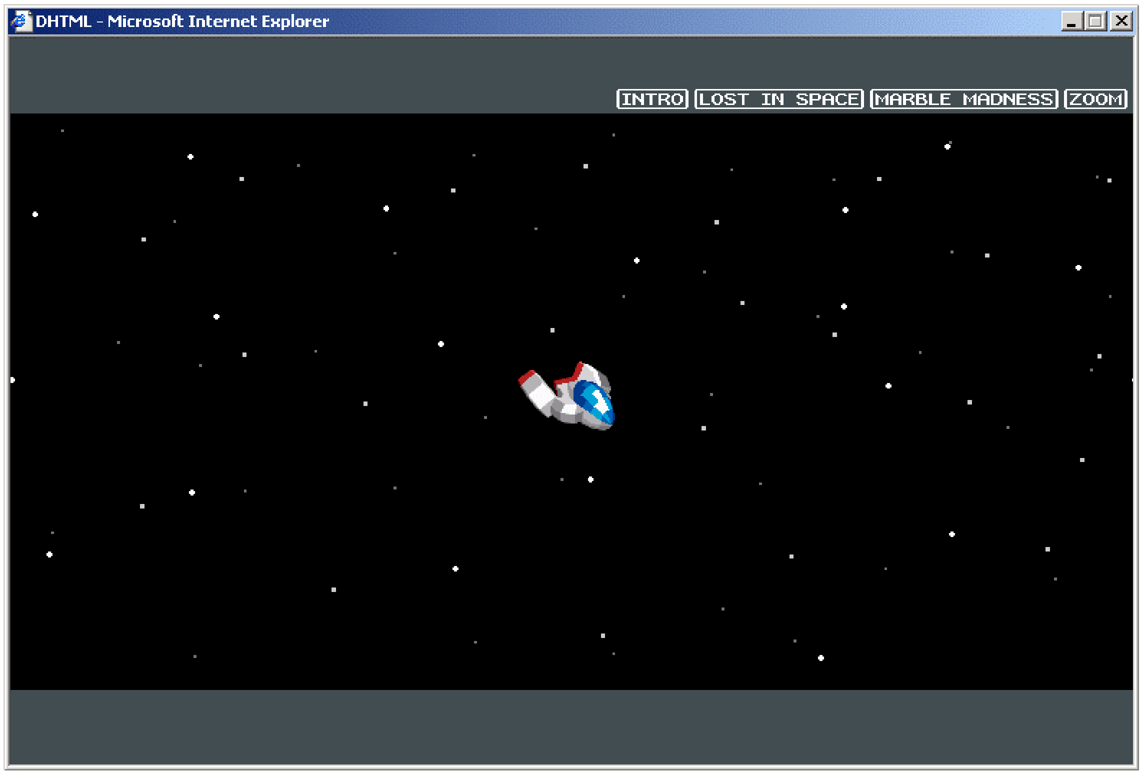


Figure 3: ‚MARBLE MADNESS'

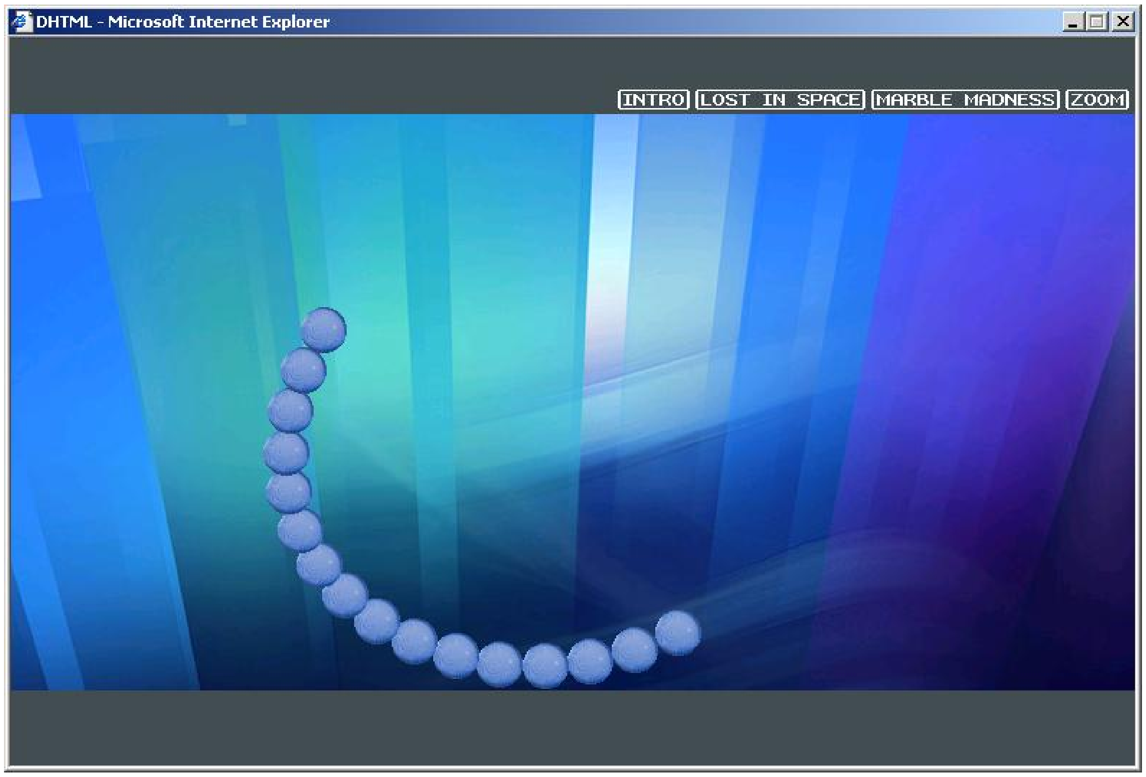
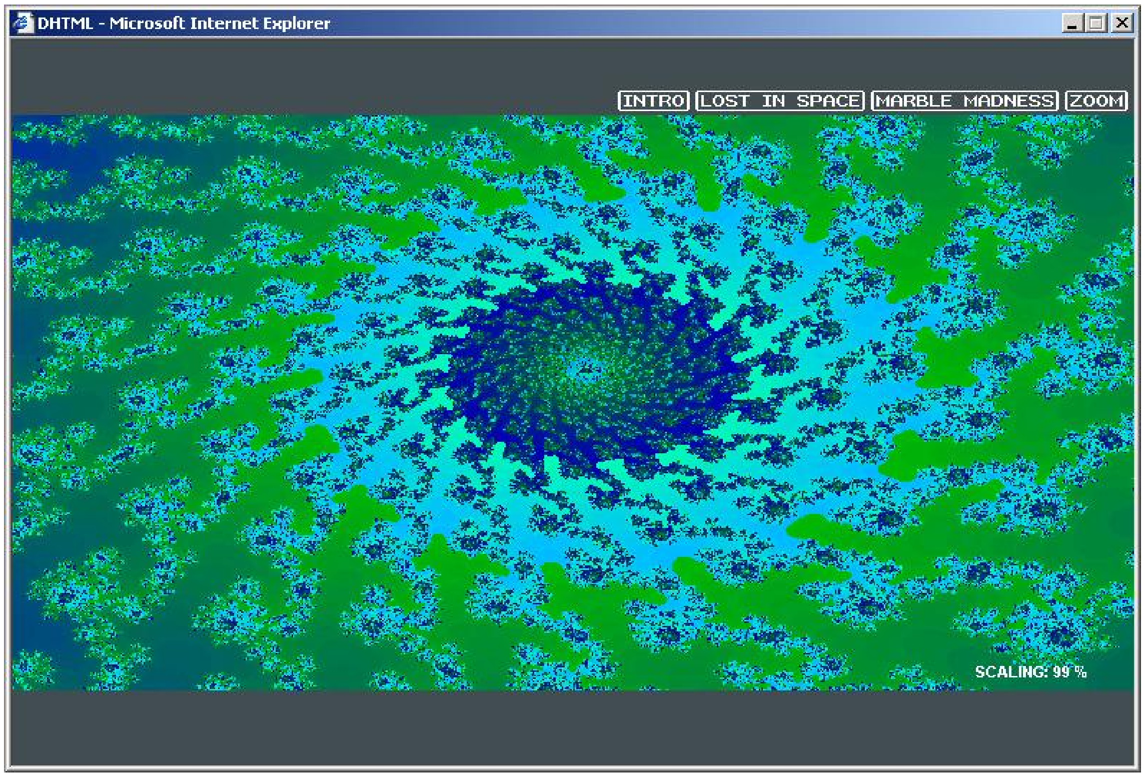


Figure 4: ‚Zoom′



## *5. Cascading Style Sheets*

**At the simplest level, a style is nothing more than a rule that tells the browser how to display a particular HTML tag. Each tag has a number of properties associated with it, whose values define how that tag is rendered by the browser. A rule defines a specific value for one or more properties of a tag. For example, most tags have a color property, the value of which defines the color used to display that tag. Other properties include font attributes, line spacing, margins and borders. With Javascript we can modify these properties dynamically.**

### 5.1 Using DIV Tags:

**When using CSS-Positioning, these properties are usually applied to the DIV (division) tag - an empty, non-formatting tag, that is best suited for CSS. When you put HTML/text into a DIV tag it is commonly referred to as one of: "DIV block", "DIV element", "CSS-layer", or just a "layer". Using DIV tag by itself has the same results as using <P></P>. But by applying CSS to DIV tags we can define where on the screen this piece of HTML will be displayed, draw squares or lines, or how to display the text that's inside it. You do this by first giving the DIV an ID.**

### 5.2 Inline Styles: The style Attribute

**The inline style is the simplest way to attach a style to a tag--just include a style attribute with the tag along with a list of properties and their values. The browser uses those style properties to render the contents of just this instance of the tag.**
**For instance, the following style tells the browser to display a layer at a absolute position, with the given properties like width, height etc.**

**Code snippet from zoom.html:**
```
<div id="param" style="position:absolute; top:380px; left:670px;" class="Text"></div>
```

**This tag is empty by now, but later it will be used to display the scaling factor of the zooming image by setting up the content dynamically with Javascript.**

## 5.3 CSS Properties

**Generally, the following properties are the ones that work (fairly closely) to the standards as defined by the W3C.**

> **position: Defines how the DIV tag will be positioned - "relative" means that the DIV tag will flow like any other HTML tag, whereas "absolute" means the DIV will be positioned at specific coordinates. Absolute positioning will be the topic of most of this tutorial.**
>
> **left:**
> **Left location (the number of pixels from the left edge of the browser window).**
>
> **top:**
> **Top location (the number of pixels from the top edge of the browser window).**
>
> **width:**
> **Width of the DIV tag. Any text/html that is inserted into the DIV will wrap according to what this value is. If width is not defined it will all be on one line.**
>
> ***Important: When using layers for animation you should always define the width. This is because in IE the default is the entire width of the screen. If you move the layer around the screen a scrollbar will appear at the bottom, which is annoying and causes the animation to slow down.***
>
> **height:**
> **Height of the DIV tag. This property is rarely needed unless you also you want to clip the layer**
>
> **clip:**
> **Defines the clipping (crop) rectangle for the layer. Makes the DIV into a precisely defined square. You define the size of the rectangle with the values of the four edges: clip:rect(top,right,bottom,left);**
>
> **visibility:**
> **Determines whether the DIV will be "visible", "hidden", or "inherit" (default).**
>
> **z-index:**
> **The stacking order of DIV tags.**

**Most of the properties are used in example pages to create moving objects.**
**For example the two logos on the intro page changes their stacking order, so sometimes they in front or behind the jumping blue bar. This is done by the function setElem() with parameter z for z-index.**

# *6. JavaScript*

**JavaScript is a scripting language built into web brows that controls HTML elements. JavaScript first appeared in Netscape 2.0, and was primarily for scripting the contents of a web page, and providing added functionality to HTML forms, frames, and windows. Netscape 3.0 added more features like image rollovers and audio/video controls. Microsoft Internet Explorer 3.0 (released shortly after Netscape 3.0) also implemented JavaScript, but marketed it as JScript which is essentially the same as JavaScript with a few minor incompatibilities that Microsoft threw in to lure developers into using their version of JavaScript.**
**Extensions to JavaScript were added in Netscape 4.0 and Internet Explorer 4.0 to give developers a way to manipulate DHTML (HTML elements that use CSS). However these extensions were not standardized before the release of the 2 browsers. And as a result we now have two versions of JavaScript that are largely incompatible.**

## 6.1 Cross-Browser JavaScript

**You can use JavaScript to access and change the properties of your CSS-P element. However, some of the syntax differs between Netscape 4.0 and Internet Explorer 4.0. By knowing where the differences are, I'll show you an easy way to create cross-browser JavaScripts - scripts that will work in both N4 and IE4.**

## 6.2 Browser Checking:

**How to check out which browser is running?**

**Code snippet from js/dhtml.js:**

```
var DHTML = 0, DOM = 0, MS = 0, NS = 0, OP = 0;

function DHTML_init() {

 if (window.opera) {
     OP = 1;
 }
 if(document.getElementById) {
   DHTML = 1;
   DOM = 1;
 }
 if(document.all && !OP) {
   DHTML = 1;
   MS = 1;
 }
if(document.layers && !OP) {
   DHTML = 1;
   NS = 1;
 }
}
```

**The document.layers object is specific to Netscape 4.0, while the document.all object is specific to IE 4.0. So by checking if the object exists we can create the boolean variables NS (for Netscape 4.0) and MS(for Internet Explorer 4.0)  etc. and assign them true or false depending on which browser is being used. Now whenever you need to check which browser someone is using you just have to use if (NS) or if (DOM):**

## 6.3 Using JavaScript and CSS-P:

**Let′s take a DIV tag from Intro.html:**

```
<div id="logo1" style="position:absolute; width:188px; height:76px; z-index:3; left: 0px; top:
-200px"><!-- top: -200 px to make it at init invisible -->
<img src="images/DhtmlLogo.gif" alt="" width="188" height="76" border="0">
</div>
```

**For Netscape the general way to access the CSS-P properties is like this:**

```
document.blockDiv.propertyName
```

**or**

```
document.layers["blockDiv"].propertyName
```

**And then for Internet Explorer it's:**

```
blockDiv.style.propertyName
```

**or**

```
document.all["blockDiv"].style.propertyName
```

**Where propertyName can be any one of left, top, visibility, zIndex, width, or any of the other CSS-P properties.**

**Here is the code from MarbleMadness.html for moving the balls(marbles):**

```
function setPos(ball,x,y){
        if (document.layers) {
                document.layers[ball].left=x+358
                document.layers[ball].top=y+184
        }
        else {
                getElem("id",ball,null).style.left = x+358
                getElem("id",ball,null).style.top = y+184
        }
}
```

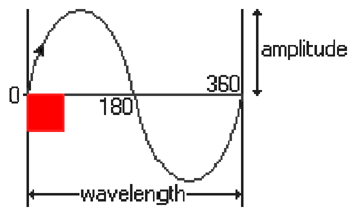
**setPos(ball, x,y) is the function to set the position of a ball, so after a refresh by the browser, it will appear at the new position.**

## 6.4 Sliding

**Sliding is just what I call an animated movement or scrolling effect. By using looping (or iterating) functions and moving the layer in small increments, you can put together any sort of animated movement you can think of. In Intro.html and MarbleMadness.html we use a simple sinus/cosinus table:**

```
var sinTab=new Array(360)

for (i=0;i<360;i++){
        sinTab[i]=parseInt(Math.sin((i*Math.PI)/wavelength)*amplitude);
}
```



## 6.5 Timing

**With the method JavaScript setTimeout()/setInterval() we can create a timing loop by using the timeout to valuate an expression after the specified timeout has occurred.**
**To call a function once use:**

```
        setTimeout("animate()",12)
```

**or**

```
        setInterval("animate()",40)
```

**to valuate an expression each time the specified interval occurs. Both functions are used to refresh the display within the given time interval.**

## 6.6 Showing and Hiding Layers

**This feature is used to hide the ‵LOADING′ message, after the page is completely loaded. Every page has the following div tag with an animated gif as content:**

```
<div id="Loading" style="position:absolute; width:188px; height:76px; z-index:90; left: 356px;
top: 196px">
<img src="images/Loading.gif"  width="67" height="8" border="0">
</div>
```

**After loading has been completed we have to hide the loading layer:**

**For Netscape**
**To hide an element in Netscape you have to use:**

```
                if (document.layers)
                        getElem("id","Loading",null).visibility = "hidden"
                else
```

**For Internet Explorer**
**To hide an element in Internet Explorer you have to use:**

```
                        getElem("id","Loading",null).style.visibility = "hidden"

                setInterval("animate()",20);
        }
```

## *7. Sources*

**Links for more DHTML information:**

**[1]** **Microsoft DHTML Documentation**
**http://msdn.microsoft.com/workshop/author/default.asp**

**[2]** **Netscape DHTML Documentation**
**http://developer.netscape.com/docs/manuals/communicator/dynhtml/index.htm**

**Links for more CSS information:**

**[3]** **W3C CSS-Positioning**
**http://www.w3.org/TR/WD-positioning.html**

**[4]** **Builder.com's CSS Guide**
**http://builder.cnet.com/Authoring/CSS/index.html**

**Links for more JavaScript information:**

**[5]** **Netscape JavaScript Guide**
**http://developer.netscape.com/docs/manuals/communicator/jsguide4/index.htm**

**[6]** **JavaScript Reference**
**http://developer.netscape.com/docs/manuals/communicator/jsref/index.htm**

**[7]** **Microsoft JScript**
**http://msdn.microsoft.com/scripting/default.htm**