

Efecte multimedia in Internet Explorer 5.5

Sabin Corneliu Buraga

Articol publicat in *PC Report*, vol.9, nr.99 - decembrie 2000

Aparut recent, navigatorul Internet Explorer 5.5 aduce citeva facilitati interesante pentru dezvoltatorii de pagini WWW, dintre care pot fi enumerate atasarea de filtre vizuale, realizarea de prezentari multimedia sincronizate sau crearea de componente hipertext cu semantici stabilite de programatorii Web, suportul pentru foile de stiluri XSL si schemele XML si altele. Despre o parte dintre acestea vom discuta in continuare.

Filtre vizuale

In Internet Explorer 5.5, creatorii de pagini Web au posibilitatea de a atasa imaginilor incluse in documentele HTML diverse efecte si filtre vizuale pe care pina acum puteau fi realizate fie static, folosind un editor grafica raster (e.g. Adobe Photoshop ori GIMP - The GNU Image Manipulation Program), fie dinamic, prin intermediul unor applet-uri Java sau script-uri JavaScript.

Intr-o forma limitata, aceste filtre existau si in Internet Explorer 4.0, dar pentru versiunile IE 5.5 si ulterioare ele au fost imbogatite si optimizate. Filtrele vor putea fi utilizate prin intermediul unei proprietati Cascading Style Sheet (CSS) - nivelul 2 denumite *filter* a carei valoare va fi un sir de caractere desemnind apelul unei functii de filtrare grafica. Sintaxa generala este urmatoarea:

```
<element style="filter:progid:DXImageTransform.Microsoft.numefiltru(proprietati)">
```

Fiind introduse prin proprietati CSS, filtrele vor putea fi atasate unui grup de elemente, folosind mecanismul foilor de stiluri, dar ele nu vor putea fi utilizate pentru palete cu mai putin de 256 de culori si pentru tag-urile `<embed>`, `<applet>`, `<object>`, `<option>`, `<select>` sau pentru elementele de tabel (i.e. `<table>` ori `<tr>`). De asemeni, filtrele vor fi ignorate daca apar specificate pentru elemente pozitionate aflate in cadrul unor elemente nepozitionate. Astfel, vom specifica inutil o proprietate *filter* pentru tag-ul `` in cadrul unei constructii de genul:

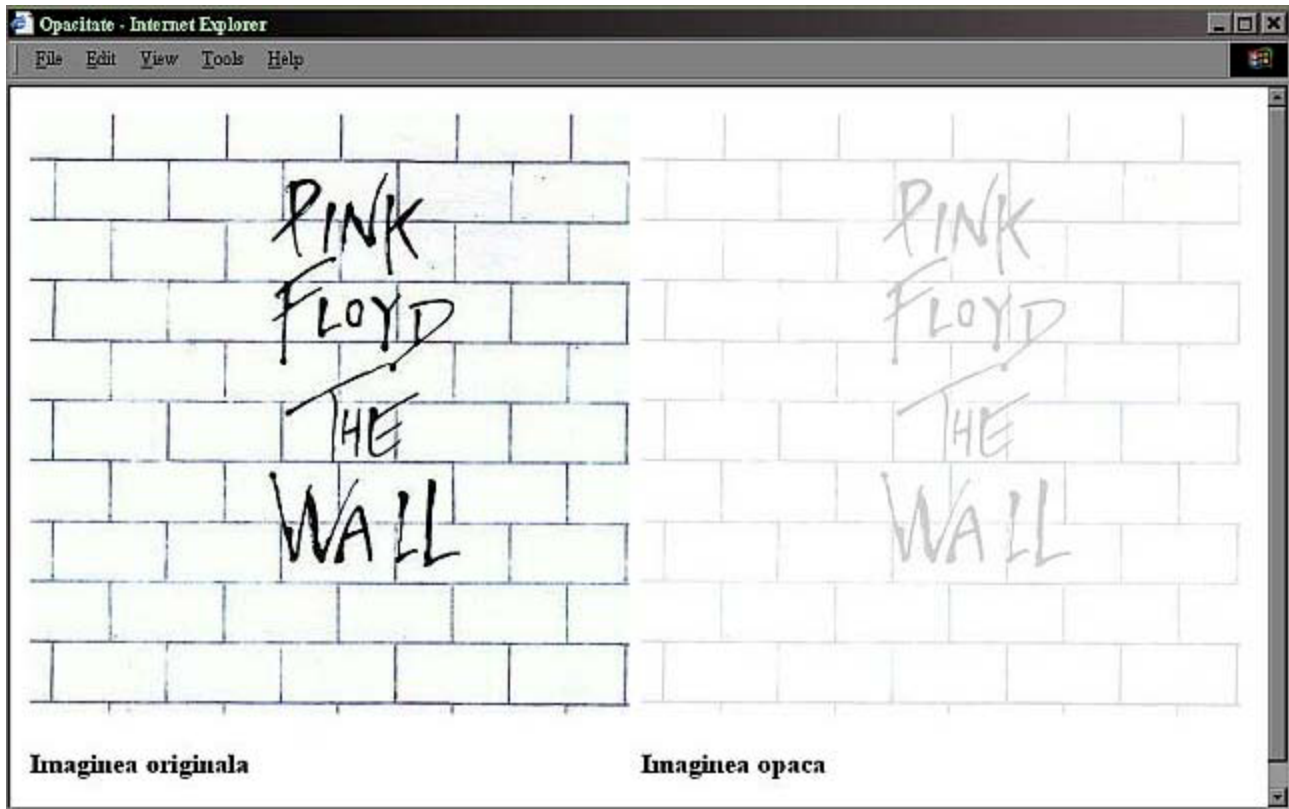
```
<div>
...
<span style="top: 30; left: 60">
...
</span>
...
</div>
```

Asadar, filtrele vor putea fi aplicate numai elementelor posedind o forma de afisare - un *layout* de afisare creat, de exemplu, prin specificarea atributelor de inaltime (*height*) si de latime (*width*) ori avind pozitionare absoluta in cadrul ferestrei navigatorului.

Iata pentru inceput setarea gradului de opacitate pentru o imagine, folosind filtrul *Alpha*:

```
<div align="center">  
  
<h5>Imaginea originala</h5>  
  
<h5>Imaginea opaca</h5>  
</div>
```

Efectul acestui filtru este prezentat mai jos:



In stanga, imaginea originala; in dreapta, imaginea opacizata

Incercati acest exemplu, inlocuind filtrul *Alpha* cu *Blur* care determina incetosarea partiala a imaginii:

```

```

Pentru a transforma (a desatura) o imagine color intr-una pe tonuri de gri vom putea folosi filtrul *BasicImage* cu parametrul `grayscale=1`. Acest filtru este responsabil si pentru realizarea de negativ asupra unei fotografii, dupa cum se poate observa din exemplul urmator:

```

```



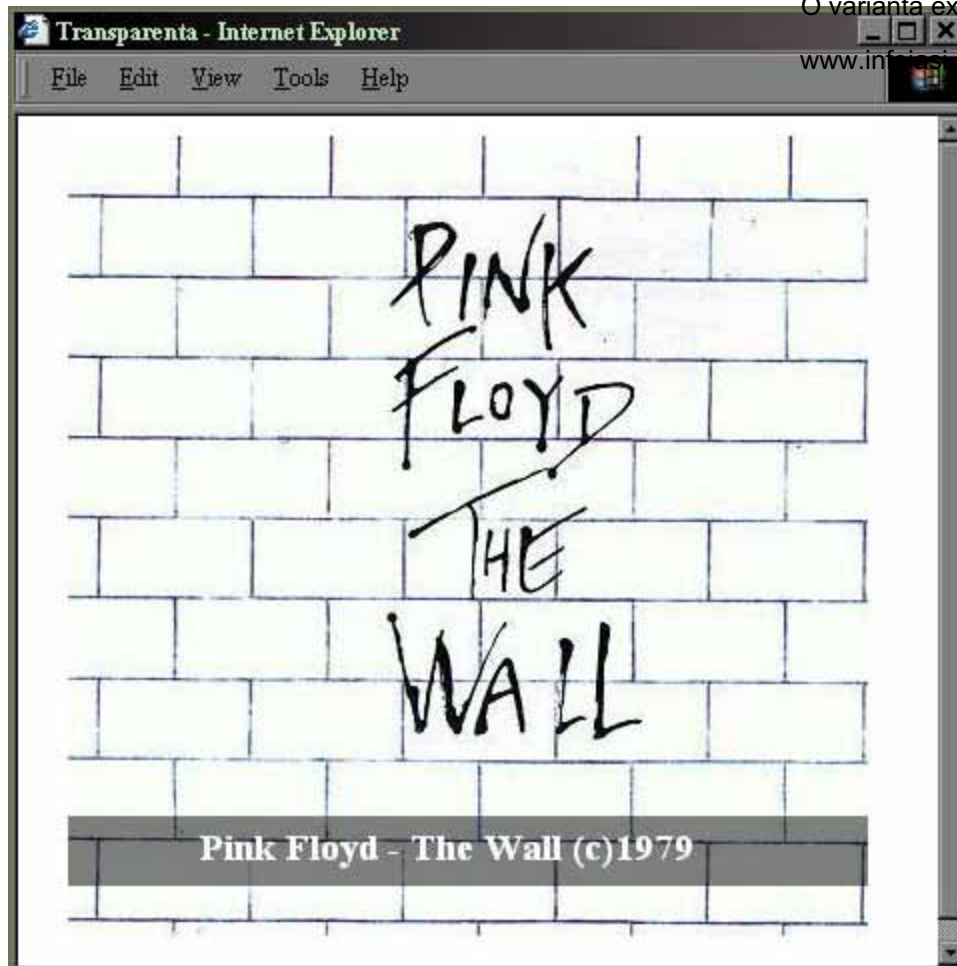
In stnga, imaginea originala; in dreapta, imaginea negativa

Incercati, de asemenea, sa aplicati *BasicImage* cu parametrul `xray=1`.

Proprietatea `opacity` a filtrului *Alpha* poate fi utila pentru suprapunerea mai multor tag-uri HTML avind diverse grade de vizibilitate, dupa cum se poate remarca si din urmatorul exemplu:

```
<!-- Coperta albumului -->

<!-- Se suprapune numele formatiei, titlul si anul albumului -->
<div style="position:absolute; top:350; left:25;
          width:400; height:35;
          background-color: black;
          filter:progid:DXImageTransform.Microsoft.Alpha(opacity=50)">
</div>
<div style="color:white; position:absolute; top:350;
          width:400; height:80; margin-top:5; margin-left:5;">
  <p style="font-size:14pt; font-weight:bold; text-align:center">
    Pink Floyd - The Wall (c)1979
  </p>
</div>
```

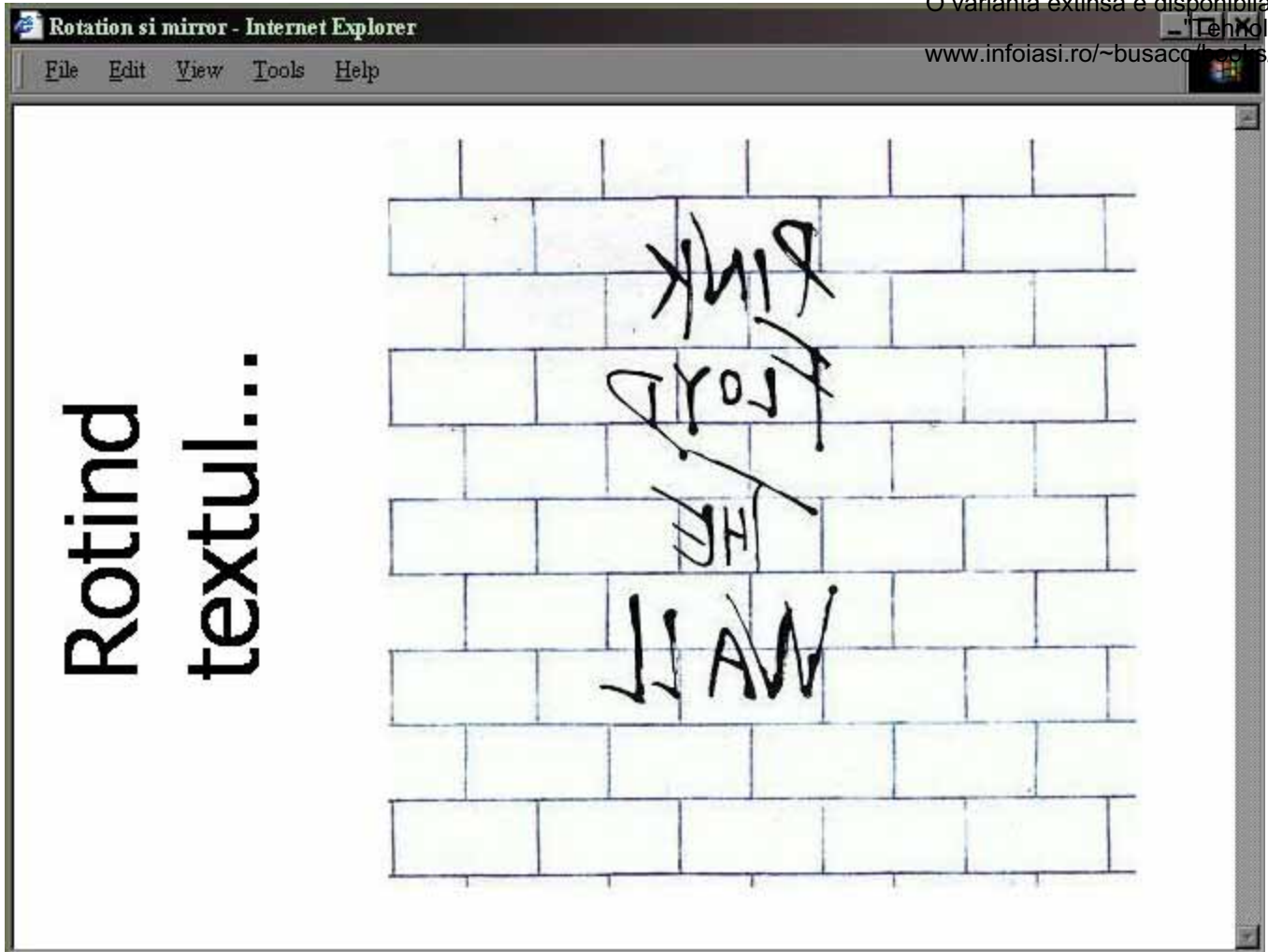


Utilizarea unor grade diferite de transparenta asupra unor elemente HTML suprapuse

Pentru efectuarea de prelucrari precum rotiri sau afisari in oglinda putem utiliza filtrul *BasicImage* impreuna cu proprietatile *rotation* si *mirror*, respectiv. Iata un exemplu de folosire pentru un text si pentru o figura:

```
<div style="height:100; width:100; font-size:40pt; font-family: sans-serif;  
    filter: progid:DXImageTransform.Microsoft.BasicImage(rotation=3) ">  
Rotind textul...  
</div>  

```



Rotirea unui text si oglindirea unei imagini

Desigur, mai pot fi folosite filtrele *Wave*, *DropShadow*, *Glow*, *Chroma*, *Light*, familiare tuturor utilizatorilor aplicatiilor de prelucrare grafica.

Iata un exemplu mai complex, constind din aplicarea mai multor filtre asupra aceleasi imagini:

```

```

Rezultatul poate fi urmarit mai jos:



In stinga, imaginea originala; in dreapta, imaginea rezultata in urma aplicarii mai multor filtre

Aplicarea dinamica a filtrelor

Pina acum am vazut diverse filtre aplicate in mod static, precizindu-se de la bun inceput diversi parametri. Pentru a realiza acest lucru in mod dinamic putem apela la obiectul `filter` predefinit de specificatia JScript pentru Internet Explorer 5.5. Pentru fiecare element HTML caruia i-am atasat filtre putem accesa colectia sa de obiecte `filters`. Iata o exemplificare:

```


<script language="javascript">
function Opac()
{
  thewall.filters.item("DXImageTransform.Microsoft.Alpha").opacity -= 10;
}
</script>

<input type="button" value="Opacizeaza" onclick="javascript:Opac()" />
```

La apasarea butonului "Opacizeaza" se va modifica proprietatea `opacity` pentru imaginea cu identificatorul `thewall`. In acest exemplu, putem observa ca putem folosi filtrele in momentul aparitiei unor evenimente.

Fiecare obiect `filter` are o serie de proprietati si metode comune tuturor filtrelor:

- `enabled` - indica daca filtrul respectiv este in actiune;
- `play()` - ruleaza filtrul respectiv (util pentru filtrele de animatie), invocind tranzitia;
- `apply()` - aplica un filtru static asupra unui element;

- `stop()` - opreste tranzitia curenta.

Iata un alt exemplu:

```
<div id="mydiv"
  style="height:100;width:200;font-size:20pt;
      filter:progid:DXImageTransform.Microsoft.Blur(strength=10)
      progid:DXImageTransform.Microsoft.CheckerBoard(duration=4)">
  Noi, formidabilii!!!
</div>

<script language="JavaScript">
// aplicarea efectului de tranzitie CheckerBoard
mydiv.filters.item("DXImageTransform.Microsoft.CheckerBoard").apply()
// Modificarea continutului elementului <div>
mydiv.innerText = "Intr-adevar?!!..."
// rularea efectului de tranzitie CheckerBoard
mydiv.filters.item("DXImageTransform.Microsoft.CheckerBoard").play()
</script>
```

Din acest exemplu, putem remarca utilizarea filtrelor si pentru text si folosirea altei categorii de filtre, asa-numitele filtre de tranzitie sau de animatie. Aceasta categorie include *CheckerBoard*, *Blinds*, *Iris*, *Barn*, *Strips*, *RandomBars* si *RandomDissolve*.

Toate aceste tranzitii pot fi utilizate in conjunctie cu filtrele vizuale, static sau dinamic.

Colectiile de filtre pot fi manipulate prin intermediul proprietatii `filter`, astfel putind adauga noi filtre la cele vechi:

```
<div id="odiv"
  style="height:150; width:250;
      filter: progid:DXImageTransform.Microsoft.Wave(strength=100)">
  Formidabili?
</div>
...
<script>
...
odiv.style.filter += "progid:DXImageTransform.Microsoft.Iris(irisstyle='STAR',duration=4)
...
</script>
```

Atunci cind modificam colectia de filtre pentru un anumit element, este recomandabil ca inainte de aceasta actiune sa dezactivam filtrele prin setarea proprietatii `enabled` ca `false`, iar apoi sa le reactivam.

Impreuna cu filtrele, poate fi utilizata si proprietatea CSS `zoom` care seteaza modul de afisare a unor elemente HTML. Astfel, putem modifica maniera de afisare a unor grupuri de tag-uri prin:

```

```

Filtrele, mai ales cele de tranzitie, pot fi utilizate pentru efectuarea de animatii similare tranzitiilor de *slide-uri* PowerPoint, asa cum se poate remarca in exemplul de mai jos:

```
<html>
```

```
<head><title>Animatii</title>
<script language="JavaScript">
var fRunning = 0

function startTrans()
{
    if (fRunning == 0) // daca filtrul nu ruleaza, se va lansa...
    {
        fRunning = 1;
        imgID.filters.item(0).apply();
        imgID.src = "final-cut.jpg";
        imgID.filters.item(0).play();
    }
}
</script>
<script for="imgID" event="onFilterChange">
fRunning = 0
</script>
</head>
<body bgcolor="white" text="black">

<h4>Apasati imaginea pentru a incepe animatia...
</body>
</html>
```

Un filtru atunci cind isi incheie activitatea sau cind tranzitia se termina va genera evenimentul `onFilterChange`. Captind acest eveniment putem aplica secvential mai multe filtre asupra aceluiasi tag HTML. Astfel, putem cicla la diferite momente de timp o serie de filtre iar la cerere sau automat sa oprim tranzitia prin intermediul metodei `stop()`.

Puteti experimenta alte tipuri de tranzitii, substituind de exemplu *Fade* cu *Blinds* si setind parametrul `Duration=3.3`.

Aceste tranzitii pot fi utilizate nu neaparat in cadrul scripturilor, ci pentru a realiza diverse efecte intre incarcarea unei pagini noi in fereastra navigatorului. Pentru aceasta, in antetul documentului HTML putem include intr-un element `<meta>` urmatoarele:

```
<meta http-equiv="Page-Enter"
    content="progid:DXImageTransform.Microsoft.Blinds(Duration=3)" />
<meta http-equiv="Page-Exit"
    content="progid:DXImageTransform.Microsoft.Barn(orientation='horizontal', motion='i
```

Prima constructie va realiza aplicarea filtrului *Blinds* la incarcarea paginii (la aparitia evenimentului `Load`), iar a doua va cauza rularea filtrului *Barn* atunci cind utilizatorul va parasii pagina (la evenimentul `Unload`). Exemplele de utilizare a acestor filtre pot fi urmarite incarcind pagina principala a site-ului Facultatii de Informatica din Iasi: <http://www.infoiasi.ro>.

In loc de "Page-Enter" si "Page-Exit" pot fi utilizate valorile "Site-Enter" si, respectiv, "Site-Exit".

Nu intotdeauna insa aceste filtre vor functiona corespunzator, probabil datorita unei scame a navigatorului.

Probleme de utilizare a filtrelor

Anumite filtre (e.g. *Shadow*, *Compositor*, *DropShadow*, *Glow* sau *Mask*) vor functiona adecvat

numai daca sint aplicate asupra unor elemente transparente. Textul fara culoare sau imagine de fundal se considera a fi transparent. Imaginile trebuie sa fie in format GIF sau avind o culoare de transparenta.

Trebuie sa avem in vedere faptul ca aplicind repetat filtre sau folosind un numar mare de filtre in cadrul paginilor Web, asupra unor imagini de dimensiuni mari, vom incetini procesul de afisare a continutului acelor documente. Este inutil sa modificam o serie de proprietati ale unui filtru dupa ce elementul avind atasat acel filtru a fost deja afisat de navigator, deoarece schimbarile nu vor putea fi observate decit reincarcind sau reimproscatind continutul acelei pagini.

Filtrele de tranzitie ruleaza asincron, deci proiectantii de pagini Web trebuie sa urmareasca fiecare stare a tranzitiilor din cadrul documentului prin intermediul evenimentului `onFilterChange` pentru a determina terminarea unei tranzitii particulare. Incarcarea imaginilor continute de o pagina Web se realizeaza tot in mod asincron, deci starea vizuala finala a acestora nu se va putea reactualiza decit atunci cind ele vor fi incarcate complet. Filtrele de tranzitie pot urma oricarui filtru static in cadrul sirului de filtre desemnat de proprietatea `filter`. Daca se doreste ca tranzitia sa fie aplicata global asupra mai multor obiecte, este mai bine ca acestea sa fie grupate intr-un element de tip `` sau `<div>` si filtrele sa fie atasate acestuia.

Filtrele prezentate mai sus sint in fapt controale DirectX incorporate in Internet Explorer 5.5, iar programatorii isi pot adauga propriile filtre urmind indicatiile *DirectX Media SDK*. Utilizarea unor filtre particulare se va face in maniera urmatoare:

```
<element style="filter: progid:Companie.tip_filtru.ume_filtru(proprietati)" >
```

Prezentari sincronizate. O introducere in HTML+TIME 2.0

Desi limbajul SMIL (prezentat intr-un dintre precedentele noastre articole) este un format portabil pentru realizarea prezentarilor multimedia, nu ofera suport de integrare in HTML. Astfel, au fost propuse o serie de extensii multimedia denumite **Timed Interactive Multimedia Extensions for HTML (HTML+TIME)** menite sa adauge HTML-ului capabilitati de temporizare, interactivitate si procesare a mediilor, urmind liniile de dezvoltare din SMIL. In prezent, HTML+TIME este suportat de Internet Explorer versiunile 4 si 5. Varianta secunda a specificatiei, HTML+TIME 2.0, inspirata din SMIL 2.0 (Boston), este implementata in Internet Explorer 5.5 sau o versiune ulterioara. SMIL 2.0 reprezinta a doua versiune a limbajului **Synchronized Multimedia Integration Language**, fiind in faza de propunere pentru standardizare la Consoritiul Web. HTML+TIME 2.0 nu ofera decit o parte dintre facilitatile enuntate de SMIL 2.0.

Suportul pentru efectuarea de prezentari sincronizate in Internet Explorer 5.5 folosind facilitatile HTML+TIME 2.0 se realizeaza prin intermediul mecanismului de atasare de comportamente elementelor HTML. Incepind cu Internet Explorer 5.0, proiectantii de pagini Web isi puteau defini propriile semantici pentru elementele HTML standard sau pentru elementele proprii, prin mecanismul de atasare de componente hipertext prin intermediul asa-numitelor *comportamente (behaviors)*. Pentru a putea utiliza HTML+TIME in cadrul navigatorului, trebuie in prealabil sa includem urmatoarele declaratii in antetul paginilor Web:

1. In primul rind este necesar sa atasam declaratia spatiului de nume HTML+TIME prin constructia:

```
<html xmlns:t="urn:schemas-microsoft-com:time">
```

O varianta extinsa e disponibila in cartea Tehnologii Web
Spatiile de nume reprezinta un mecanism XML pentru evitarea ambiguitatilor de utilizare a diferitelor elemente XML avind aceleasi denumiri dar semantici diferite.
http://www.iasi.ro/~busaco/books/web.htm

- Urmeaza apoi sa declaram comportamentele HTML+TIME 2.0 (`time2`) in cadrul unui element `<style>` in maniera urmatoare:

```
<style>
  .time { behavior: url(#default#time2); }
</style>
```

Pentru a folosi specificatia HTML+TIME 1.0 (considerata in prezent demodata), putem utiliza `time` in loc de `time2`.

De notat faptul ca aceste comportamente pot fi atasate sau eliminate si prin intermediul metodelor JScript `addBehavior()` si, respectiv, `removeBehavior()`.

- Mai trebuie sa adaugam o declaratie de importare a comportamentelor `time2` prin instructiunea de procesare XML:

```
<?IMPORT namespace="t" implementation="#default#time2">
```

Exemple

- Sincronizarea aparitiei unor elemente se poate realiza in modul urmatoar:

```
<html xmlns:t="urn:schemas-microsoft-com:time">
<head>
  <style>
    .time { behavior: url(#default#time2); }
  </style>
  <?IMPORT namespace="t" implementation="#default#time2">
</head>
<body>
  <p>Vor urma citeva linii mai jos...</p>
  <!-- Imaginea va fi afisata dupa 2 secunde -->
  
  <p class="time" begin="4" dur="5">...dupa 4 secunde.</p>
  <p class="time" begin="6" dur="5">...dupa 6 secunde.</p>
  <p>Aceasta este ultima linie de text.</p>
</body>
</html>
```

Folosind pentru elementele HTML dorite clasa `time` prin constructia `class="time"` le atasam comportamentul temporal dat de specificatia HTML+TIME 2.0.

- Utilizarea, intr-un mod mai complex, a atributelor `begin` si `dur` conduce la aparitia intermitenta a unor elemente HTML ca in exemplul de mai jos:

```
<table border="1" id="clipitor" class="time"
  begin="0; clipitor.end+2" dur="1">
<tr>
<td>
Acest tabel va apare intermitent!
</td>
</tr>
</table>
```

Astfel, posedam un mecanism mai sofisticat de clipire, aproape similar controversatului tag `<blink>` din Netscape.

O varianta extinsa e disponibila in cartea Tehnologii Web
www.infoiasi.ro/~busaco/books/web.htm

Pentru atributul `begin` s-au precizat doua valori, prima fiind 0 (insemnind imediat dupa incarcarea paginii), iar a doua (`clipitor.end+2`) fiind relativa la timpul de terminare a afisarii elementului cu identificatorul `clipitor`. Pot fi specificate valori temporale multiple pentru `begin`, ca de exemplu `<p begin="1;5;9;33" dur="2" class="time">...</p>`. Astfel, paragraful va apare pentru 2 secunde in prima secunda, apoi in a cincea secunda si asa mai departe. Aceste constructii functioneaza si pentru atributul `end`:

```
<h4 id="dali" class="time" begin="0; dali.end+2" dur="4">
Unicornul lui Salvator Dali...
</h4>

```

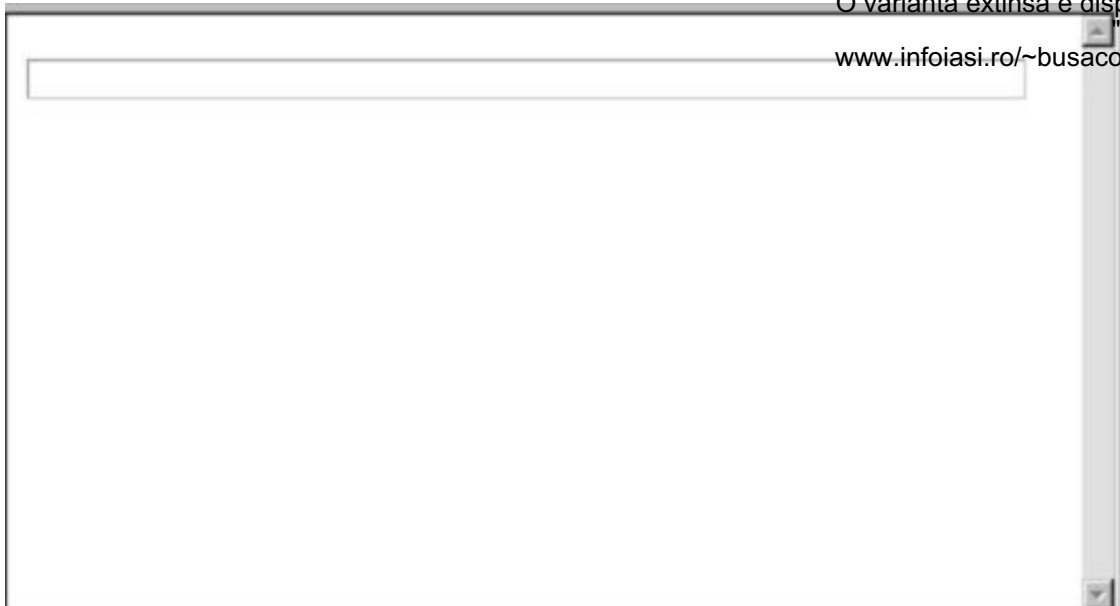
Astfel, putem observa ca exemplul ne ofera posibilitatea de a preciza sfirsitul sincronizat al afisarii a doua elemente de tip text si respectiv imagine. Desigur, pot fi imaginat si scenarii sincronizate sofisticate pentru a conferi paginilor Web actuale o dinamicitate sporita.

c. HTML+TIME 2.0 permite, de asemeni, conceperea de prezentari multimedia sincronizate interactive, in functie de anumite evenimente generate de utilizator. Putem preciza ca afisarea elementelor HTML sa se realizeze la apasarea butonului sting al mouse-ului asa cum se poate remarca din exemplul de mai jos:

```
<hr size="20" width="500" align="left" id="hr_clic" />
<table class="time" begin="hr_clic.click" dur="10"
border="1" width="500">
<tr>
<td>
<h5 align="center">
Acest tabel va apare dupa ce se apasa pe bara orizontala<br />
si va fi afisat 10 secunde...
</h5>
</td>
<td>

</td>
</tr>
</table>
```

Efectul acestui exemplu poate fi urmarit in figura urmatoare care infatiseaza afisarea tabelului dupa apasarea barei orizontale din partea superioara a ferestrei navigatorului (inainte si dupa efectuarea *click*-ului cu mouse-ul:



Interactivitatea cu utilizatorul in HTML+TIME 2.0

d. Pentru utilizarea unor obiecte multimedia se pot folosi urmatoarele tag-uri HTML+TIME 2.0, similare celor din SMIL: `<media>`, `<video>`, `<audio>`, ``, `<animation>` si `<ref>`. De exemplu, pentru a starta si a opri la cerere un video-clip putem scrie:

```
<button id="start">Start</button>
<button id="stop">Stop</button>
<t:media begin="start.click" end="stop.click"
  src="clock.avi" mute="true"
  timeAction="display"
  style="height=200; width=250" />
```

e. Functionalitatea elementului `<switch>` poate fi intrevazuta in urmatorul exemplu:

```
<t:switch>
<span class="time" systemLanguage="ro">Româna</span>
<span class="time" systemLanguage="es">Español</span>
<span class="time" systemLanguage="pt">Portuguese</span>
<span class="time" systemLanguage="en">English</span>
```

```
<span class="time">Nici una din cele testate...</span>
</t:switch>
```

O varianta extinsa e disponibila in cartea
"Tehnologii Web"
www.infoiasi.ro/~busaco/books/web.htm

Putem realiza prezentari multimedia complexe prin imbinarea utilizarii filtrelor prezentate mai sus cu elementele de sincronizare oferite de HTML+TIME 2.0.

Componente hipertext: un mod de extindere a elementelor HTML

Spuneam mai sus ca designerii de pagini Web isi pot defini propriile tag-uri intr-o maniera asemanatoare XML-ului. Definirea unei componente noi se realizeaza prin intermediul constructiei urmatoare, scrisa intr-un fisier `.htc` (HyperText Component file) stocat la client sau pe un server Web (de regula suportind ASP):

```
<PUBLIC:COMPONENT NAME="componenta">
<PUBLIC:ATTACH event="eveniment" handler="functie" />
...
<PUBLIC:PROPERTY name="proprietate" />
<PUBLIC:METHOD name="metoda" />
<SCRIPT>

// codul scriptului care implementeaza metoda
// si toate functiile care trateaza evenimentele specificate

</SCRIPT>
</PUBLIC:COMPONENT>
```

Exemplu

Urmatorul exemplu, rulind sub Internet Explorer 5 sau superior, ilustreaza adaugarea unui comportament la un tag definit de utilizator pentru realizarea efectului de clipire a elementului `<blink>` prezent in Netscape Communicator.

Vom defini o componenta stocata in fisierul `blink.htc` a carei actiune va fi realizarea efectului de clipire:

```
<PUBLIC:COMPONENT name="blink">
<PUBLIC:PROPERTY name="speed" />
<PUBLIC:METHOD name="doBlink" />
<PUBLIC:ATTACH event="oncontentready" handler="Init" />
<SCRIPT>
// codul JavaScript pentru clipire
// adaptat dupa InsideHTML.com
// metoda de clipire
function doBlink() {
    this.style.visibility = this.style.visibility == "" ? "hidden" : ""
}

// functia de initializare apelata la aparitia
// evenimentului <i>oncontentready</i>
function Init() {
    if (!speed) // daca nu-i specificata viteza de clipire...
        speed = 1000 // atunci va clipi din secunda in secunda
    setInterval(uniqueID + ".doBlink()", speed)
}
</SCRIPT>
</PUBLIC:COMPONENT>
```

In cadrul documentului Web, va trebui sa atasam un comportament elementului `<blink>` constructia:

```
<style>
<!-- ascundere de browserele mai vechi
  @media screen /* clipire doar pentru ecran */
  {
    blink { behavior:url(blink.htc); }
  }
-->
</style>
```

Putem specifica, prin atributul `speed`, si viteza de clipire definita in cadrul componentei ca proprietate. Astfel, in corpul paginii, vom scrie:

```
<blink speed="2000">
  Acest text clipeste cu viteza de 2 secunde...
</blink>
```

La fiecare `<blink>` va fi executata metoda JavaScript definita in fisierul `blink.htc`.

Listing-ul complet al paginii Web utilizind tag-ul `<blink>` este dat mai jos, unde am specificat propriul nostru spatiu de nume `bl`:

```
<html xmlns:bl="urn:schemas-microsoft-com">
<head>
<style>
<!-- ascundere de browserele mai vechi
  @media screen /* clipire doar pentru ecran */
  {
    blink { behavior:url(blink.htc); }
  }
-->
</style>
<?IMPORT namespace="bl" implementation="blink.htc" ?>
</head>
<body>
<bl:blink speed="2000">
  Acest text clipeste cu viteza de 2 secunde...
</bl:blink>
</body>
</html>
```

In acest mod putem defini propriile noastre elemente pe care sa le folosim ulterior in cadrul paginilor Web.

Daca in Internet Explorer 5.0, componentele hipertext erau convertite in documente HTML interne care apoi erau procesate de navigator, conducind la diminuarea vitezei de incarcare, in Internet Explorer 5.5 acestea pot fi declarate "usoare", navigatorul operind o serie de optimizari interne in vederea procesarii mai rapide. La fel se intimpla si cu mecanismul de cadre (*frames*) care a fost rescris in intregime.

Concluzii

Am putut vedea in prezentul material o serie dintre noutatile aduse de Internet Explorer 5.5, mai ales in domeniul hipermediei, utile pentru designerii Web care pot beneficia, de acum incolo, de facilitati care pina acum erau posibile doar utilizind aplicatii externe (e.g. Macromedia Flash) sau script-uri sofisticate.