Pedagogická Fakulta Jihočeské Univerzity

Katedra fyziky

DHTML

DIPLOMOVÁ PRÁCE

Tomáš Blaťák

vedoucí diplomové práce

PaedDr. Petr Pexa

České Budějovice 2000

Anotace

Práce se zabývá základy DHTML. Je soustředěna na popis dynamických prvků HTML, se zaměřením na jejich objektový model a strukturu. Dále se práce zabývá některými praktickými ukázkami DHTML. Příklady jsou nejnovějšími
možnostmi DHTML, z čehož plyne omezení, spouštět aplikace v Internet Exploreru verze 5.0, pro nějž jsou vytvořeny.

Prohlašuji, že jsem tuto diplomovou práci vypracoval samostatně, a že jsem veškerou použitou literaturu uvedl v Seznamu odborné literatury.

Tomáš Blaťák

OBSAH

1. ÚVOD

V současné době je internet běžným prostředkem komunikace mezi lidmi. Je již přirozeným médiem v zprostředkování a šíření informací. Internetové stránky se stávají velmi populárními snad ve všech oborech. Internet je zahlcen bezpočtem prezentací firem, škol, institucí, organizací a spolků. Pracuje v něm obrovský počet vyhledávacích serverů pro lepší navigaci, můžete navštívit spousty diskusních skupin a klubů, vybrat si z neomezeného počtu zboží ve virtuálních obchodech, či si objednat letní dovolenou. Zkrátka možnosti internetu jsou takřka neomezené.

Proto není divu, že ve vytváření internetových stránek je skryt nový počítačový obor. V dnešní době již není problém si vytvořit vlastní internetové stránky, pomocí různých aplikačních programů, které mnohdy nevyžadují ani znalost základů HTML. Proto tvorbu stránek zvládají i děti předškolního věku.

Pokud však chceme, aby naše stránky vypadaly profesionálně a šly tzv. s dobou, je potřeba znát nejen základy HTML (Hypertext Markup Language), ale i jeho podrobnější popis, se spousty odvozených programovacích jazyků a aplikací. Nestačí však být pouze dobrým programátorem. Je třeba také zvládat základní znalosti práce s grafikou a projevit pro ni cit.

Dnes je standardem ve vytváření internotových stránek dynamické HTML. V následující části si ukážeme na jakém principu je DHTML založeno a dále bude pokračováno spoustou příkladů, ukazujících nové metody, které DHTML přineslo do programování internetových stránek.

2. DHTML

2.1 Co je DHTML

Jde o nové rozšíření programovacího jazyka HTML (Hypertext Markup Language), sloužící zejména pro dynamickou prezentaci dokumentu. Velkou výhodou DHTML je to, že do samotného jazyka HTML nepřidává nic nového. Jen rozvíjí stávající možnosti pomocí stylů a scriptů. Každý prvek je schopen reagovat na jakékoliv události v rámci stránky (pohyb myši, “klik”, “double-klik”, volba ve formuláři apod.), a to prakticky i bez skriptování.

2.2 Z čeho se skládá DHTML

DHTML se skládá z CSS verze 2, HTML verze 4 a skriptovacích jazyků. Na DHTML není nic nového, ale využívá možnosti těchto prostředků k tomu, aby dodalo stránce dynamický rozměr.
HTML v 4 a CSS v 2 jsou spolu a vlastně i s JavaScriptem a VBScriptem integrovány. To co bylo potřeba upravit, byla zpětná vazba ze skriptovacího jazyka na CSS.

2.2Vlastnosti

2.2.1 Dynamické styly

Popis stylu dokumentu může být kdykoliv změněn. Dokument nemusí znovu komunikovat se serverem, ani nemusí být znovu zaveden z vyrovnávací paměti.
Vše je navrženo tak, aby bylo umožněno jakékoliv zobrazení změny a to okamžitě.

2.2.2 Pozicování

Dané prvky lze pozicovat (pomocí souřadnic x, y) v různých vrstvách (rovinou z), což umožňuje přesné rozmístění objektů na stránce, překrývání objektů a manipulaci s nimi. Kombinace dynamických stylů, pozicování, průhledných obrázků a transparentních řídicích prvků nabízí širokou škálu animačních voleb.

2.2.3 Dynamický obsah

Objektový model DHTML umožňuje pracovat s obsahem dokumentu a měnit ho. Stejně jako u dynamických stylů, není třeba aby byl dokument znovu zaveden. Prvky na stránce lze rušit, vkládat, modifikovat (např. text), skripty mohou konstruovat a měnit obsah stránky za chodu. (např. tikající hodiny).

2.2.4 Podpora

Internet Explorer 4.0 podporuje poslední normu HTML 4.0, CSS1 a spoustu dalších rozšíření CSS. Tyto normy definují to, co je objektovým modelem dynamického HTML nechráněno.

2.2.5 Přístup ke struktuře

Všechny prvky dokumentu jsou přístupné prostřednictvím objektového modelu DHTML. K vnitřním ovládacím prvkům bylo přidáno rozšíření pro HTML a CSS, což umožňuje, aby se s těmito ovládacími prvky pracovalo v tlačítkách a ovládacích prvcích textu.

2.2.6 Multimédia

V Internet Exploreru 4.0 jsou úzce spojeny multimediální efekty s obsahem dokumentu. Efekty obsahují filtry, takže mohou napodobovat například světelné zdroje a stíny.

2.4 Objektová hierarchie DHTML

Objektová hierarchie dynamického HTML (tab. č. 1) je aplikační programové rozhraní pro vytváření živých a interaktivních stránek. Objekty v hierarchii představují prohlížeč a prvky stránky HTML.

WINDOW

- **location**
- **frames**
- **history**
- **navigator**
- **event**
- **screen**
- **document**
  - **all**
  - **anchors**
  - **applets**
  - **body**
  - **forms**
  - **frames**
  - **images**
  - **links**
  - **selection**
  - **scripts**
  - **styleSheets**

tab. č.1

2.5 Window

Objekt *window* spravuje informace o prohlížeči a existuje tak dlouho, jako existuje okno aplikace prohlížeče. Objekt *window* je v objektovém modelu HTML objektem nevyšší úrovně, nemusí být tedy při přístupu k vlastnostem okna explicitně odkazován.

2.6 Event

- objekt *event* zpřístupňuje parametry události a umožňuje ovládání prostupování události a předdefinované akce.

Vlastnosti:

event.srcElement	vrací prvek, který událost generuje jako první (jestliže například klikneme na obrázek, který je odkazem, je obrázek vlastností srcElement, zatímco událost prostupuje přes kotvu, tělo a dokument
event.cancelBubble	provádí zastavení prostupování události vzhůru v hierarchii, standardně je *false* a událost prostupuje vzhůru, *true* zastaví prostupování aktuální události
event.returnValue	přepisuje předdefinované akce události vzhůru v hierarchii

Objekt event zpřístupňuje typ události prostřednictvím vlastnosti *type,* jenž vrací název dané události, napsaný malými písmeny bez předpony *on. (onmmousedown* vráceno *mousedown)*

Souřadnice myši:

clientX, clientY související	svislá a vodorovná souřadnice ukazatele myši, s klientskou oblastí okna
offsetX, offsetY související	svislá a vodorovná souřadnice ukazatele myši, se zobrazovaným kontextem
screenX, screenY související	svislá a vodorovná souřadnice ukazatele myši, s obrazovkou

Klávesa a tlačítko myši:

button	aktuální nastavení stisknutých tlačítek myši: 0 - není stisknuto žádné tlačítko 1 - je stisknuto levé tlačítko myši 2 - je stisknuto pravé tlačítko myši 4 - je stisknuto prostřední tlačítko myši Parametr *botton* představuje kombinovaný stav všech tlačítek myši.
ctrlKey	logická hodnota, udávající stisknutí klávesy *Ctrl*
altKey	logická hodnota, udávající stisknutí klávesy *Alt*
shiftKey	logická hodnota, udávající stisknutí klávesy *Shift*

Události myši:

Slouží k sledování různých stavů myši, včetně každého pohybu myši po prvku a z prvků, stejně jako stav při stisku tlačítek myši.

onmousedown	bylo stisknuto tlačítko myši
onmousemove	myš se pohybuje nebo se právě pohnula
onmouseup	tlačítko myši bylo uvolněno
onclick	bylo stisknuto levé tlačítko myši nebo byla vyvolána předdefinovaná akce
ondblick	bylo poklepáno levým tlačítkem myši, v časovém rozsahu který je v daném systému nastaven
onmouseover	ukazatel myši se dostal do oblasti prvku
onmouseout	ukazatel myši opustil oblast prvku
ondragstart	byla spuštěna akce táhni a pusť
onselectstart	myší byl vyvolán nový výběr prvku
onselect	probíhá výběr

Události klávesnice:

DHTML přidává tři události pro sledování úhozů uživatele: *onkeydown, onkeyup* a *onkeypress,* které se spouštějí v tomto pořadí. Onkeydown a onkeyup se spouští pokaždé, když je klávesa na klávesnici stisknuta a poté uvolněna. Onkeypress se spustí po stisku libovolné klávesy ANSI.

Objekt *event* má čtyři vlastnosti pro určení stavu, v jakém se klávesnice nachází při uskutečnění dané události. *CtrlKey, altKey ,shiftKey* stejné jako vlastnosti pro události myši.

Vlastnosti:

keyCode	hodnota ASCII stisknuté klávesy, nastavení této vlastnosti na nulu v rutině události *onkeypress* událost zruší.
shiftKey	stav klávesy Shift (*true/false*)
altKey	stav klávesy Alt (*true/false*)
ctrlKey	stav klávesy Ctrl (*true/false*)

2.7 Location

- rozděluje text adresy na snadno zpracovatelné komponenty. Vlastnosti vztahující se k URL:

protocol://hostname:port/pathname?search#hash

Všechny URL osahují *protocol, hostname a pathname.* Vlastnosti *search, hash a port,* nemusejí mít hodnoty spojené s nimi.

search - představuje řetězec, který zadává server ve skriptech CGI

hasch - reprezentuje záložku na stránce

Metody objektu:

reloud[force] - volání *reloud* je stejné jako klepnutí na tlačítko obnovit v prohlížeči; vyvolá se tím opakované zavedení celé stránky, byla-li tato stránka změněna, jestliže uvedeme hodnotu *true* v parametru force vynutíme tím znovunačtení stránky aniž by byla změněna.

replace - navádí k nové stránce, pracuje podobně jako *href,* ale s tím že nepřidává stránku do seznamu historie; metoda může sloužit například jako přesměrování URL na straně klienta.

2.8 Navigator

- odkazuje na objekt obsahující informace o klientovi. Při použití tohoto objektu může být program proveden v přímé závislosti na typu prohlížeče a číslu jeho verze.

Vlastnosti:

appCodeName	kód, který je přiřazen jménu (např. Mozzila)
appVersion	vrací hodnoty prohlížeče *platform*, *information a extraInformation.*
Další:	
	userAgent, appName
Nové:	
cpuClass	typ CPU, hodnoty pro PC s Pentiem je x86
systemLanguage	výchozí jazyk systému, pro americkou angličtinu je to hodnota *en-us*
userLanguage	výchozí jazyk uživatele, pro americkou angličtinu je to hodnota *en-us*
platform	aktuální operační systém uživatele
appMinorVersion	vedlejší verze aplikace prohlížeče, pro IE 4.0 je to hodnota nula
onLine	hodnota typu Boolean, jenž určuje, zda uživatel čte stránku on-line

Vlastnost *appVersion vrací* verzi klienta ve formátu:

clientVersion (platform; imformation;[extraInformation])

platform	určuje platformu na které běží prohlížeč
imformation	úroveň šifrování produktu
extraInformation	vrací platformu nebo číslo požadovaného operačního systému

2.9 Screen

- obsahuje informace o aktuálním uživatelském zobrazení, včetně rozlišení obrazovky a nastavení barev. Neboli umožňuje analyzovat podporu obrazu u klienta a podle získaných informací zobrazení aktualizovat.

Vlastnosti:

width	vodorovné rozlišení obrazovky v pixelech
height	svislé rozlišení obrazovky v pixelech

colorDepth	počet bitů na pixel, používaný zobrazením nebo cache
availHeigh	výška obrazovky uvnitř umístěných oken (bez ukotvených panelů)
availWidth	šířka obrazovky uvnitř umístěných oken (bez ukotvených panelů)

Tyto vlastnosti mohou sloužit například jako přesměrování klienta s nízkým rozlišením obrazu, do náhradního dokumentu.

2.10 Frames

Soubor všech window objektů definovaných daným dokumentem.

2.11 History

Navštívené URL. Objekt *history* má tři metody *go, forward* a *back.* Vlastnost lenght zpřístupňuje řadu prvků ze seznamu historie.

2.12 Document

Vlastnost *document* vrací objekt *document* , který představuje stránku, která je právě obsažena v okně. Přes vlastnost *document* může být získán přístup ke stylu, struktuře a obsahu dokumentů. Je nejdůležitějším a nejvýkonnějším objektem objektového modelu DHTML. Na jednotlivé objekty se odvoláváme pomocí tečkové notace. Například : *window.document.body*

Změna barev v dokumentu:

Objekt *document* obsahuje vlastnosti, které nám definují barvu textu, pozadí a odkazů

vlastnost:	atribut:
alinkColor	ALINK
bgColor	BGCOLOR
fgColor	TEXT
linkColor	LINK
vlinkColor	VLINK

- vlastnost *bgColor* řídí barvu pozadí dokumentu a vlastnost *fgColor* řídí předdefinovanou barvu textu. Tři vlastnosti barev odkazů představují barvy aktivního, navštíveného a nenavštíveného odkazu.

Velikost souboru:

Dokument dává k dispozici vlastnost *fileSize*, jenž vrací původní velikost dokumentu v kilobajtech. Vrácená hodnota představuje počet bajtů v souboru, které byly zavedeny, bez vkladu scriptů.

Datum:

fileCreatedDate	kdy byl dokument poprvé vytvořen
fileModifiedDate	kdy byl naposledy aktualizován
fileUpdateDate	kdy byl naposledy zaveden do rychlé vyrovnávací paměti

Další vlastnosti dokumentu:

title	titulek dokumentu
location, URL	umístění zdroje
mimeType	u dokumentů HTML vrací hodnotu *Internet Document*

Cookies:

Stránka HTML umí uložit na počítači klienta malý soubor dat ve speciálním souboru. Tento soubor se nazývá cookie.

Vlastnosti:

cookie používá hodnotu řetězce v následujícím formátu:

name=*hodnota*; [expires=*datum*; [path=*cesta*; [domain=*doména* [secure]]]]

Jediným povinným parametrem je název-hodnota. Cookie lze výhodně využít například pro počítání přístupů na webovou stránku.

Dostupnost dokumentu:

Při běžném zavádění prochází dokument nebo objekt čtyřmi stavy:

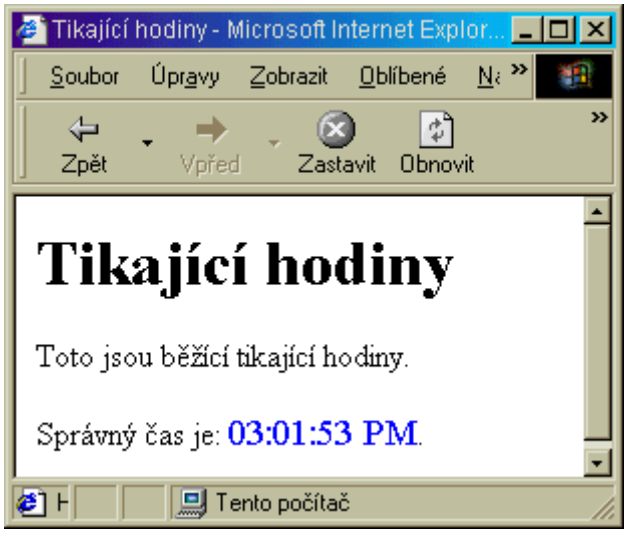
unitialized	stránka nebo objekt není ještě spuštěná; hned jak se spustí zavádění, stav ihned přejde do stavu loading (zaváděný)
loading	stránka nebo objekt jsou právě zaváděny
interaktive	se stránkou nebo objektem lze pracovat aniž by byl ještě

	zaveden
complete	stránka nebo objekt jsou právě zavedeny

3.Praktická část

3.1 Běžící hodiny

Dosud bylo možné přidat běžící hodiny pouze pomocí apletů, obrázků s komplexním kódem apod. Tyto běžící hodiny běží přímo v dokumentu. Program specifikuje, že výstup hodin bude pomocí ID **Clock** umístěn uvnitř prvku Span. Po každém tiknutí hodin bude obsah prvku Span nahrazen novým časem. (viz. obr. 1)



obr. 1

```
<HTML>
  <HEAD>
    <TITLE>Tikající hodiny</TITLE>
    <STYLE TYPE="text/css">
      #clock {color:blue; font-size:120%}
    </STYLE>
    <SCRIPT LANGUAGE="JavaScript">
      var MS = navigator.appVersion.indexOf("MSIE");
      window.isIE4 = (MS > 0) &&
```

```
        (parseInt(navigator.appVersion.substring(MS + 5, MS + 6))
          >= 4);
      function lead0(val) {
        return (val < 10) ? "0" + val.toString() : val;
      }
      function buildTime() {
        var time = new Date();
        var ampm = "AM";
        var h = time.getHours();
        if (h > 12) {
          h = h - 12;
          ampm = " PM";
        }
        return lead0(h) + ":" + lead0(time.getMinutes()) + ":" +
          lead0(time.getSeconds()) + ampm;
      }
      function tick() {
        // Replace the clock's time with the current time.
        document.all.clock.innerText = buildTime();
      }
    </SCRIPT>
  </HEAD>
  <BODY ONUNLOAD="if (null != window.tmr)
clearInterval(window.tmr);"
      ONLOAD="if (window.isIE4)
        window.tmr = setInterval('tick()', 999);">
    <H1>Tikající hodiny</H1>
    <P>Toto jsou běžící tikající hodiny
    </H1>
    <P>Správný čas je:
    <SPAN ID="clock">
      <SCRIPT LANGUAGE="JavaScript">
        document.write(buildTime());
      </SCRIPT>
    </SPAN>.
  </BODY>
</HTML>
```

3.2 Automaticky rolující okno

Program ukazuje použití metody **Scrool.** Metodu *Scrool* použijeme k rolování textu na požadované místo pomocí souřadnic x,y (ty jsou zadány v pixelech vůči levému hornímu rohu dokumentu. (viz. obr. 2)

scrollBy(offset horizontal, vertikal)	roluje dokument o zadané úseky, v pixelech
scrollTo(offset horizontal, vertikal)	roluje dokument na zadanou pozici. V pixelech, *scrollTo* a *scroll* jsou navzájem zaměnitelné
scroll(offset horizontal, vertikal)	

obr. 2

```
<HTML>
  <HEAD>
    <TITLE>Automaticky se rolující stránka</TITLE>
    <SCRIPT LANGUAGE="JavaScript">
      var tScroll;
      var curPos = 0;
        function runTimer() {
        curPos = document.body.scrollTop + 3;
        window.scroll(0, curPos);
```

```
        // Spustí se když dokument dosáhene konce
        if (curPos > document.body.scrollHeight -
            document.body.clientHeight)
          window.scroll(0, 0);
        tScroll = window.setTimeout("runTimer();", 100);
      }
      window.onload = runTimer;
      window.onunload = new Function("clearTimeout(tScroll)");
    </SCRIPT>
  </HEAD>
<BODY STYLE="margin-bottom:350pt">
    <H1>automaticky se rolující okno</H1>
    <P>První řádek.
    <P>Druhý řádek.
    <P>Třetí řádek.
    <P>text se automaticky
    <P>roluje od dola vzhůru,
    <P>potom opět to samé.
    <P>Další řádek.
    <P>Předposlední  řádek.
    <P>Poslední řádek rolující stránky.
</BODY>
</HTML>
```

3.3 Obrazová mapa

Obrazové mapy nám definují různé oblasti na nichž klepneme myší. Když klient klepne na určitou oblast v obrázku, je předdefinovanou akcí navedení uživatele na určitou stránku.

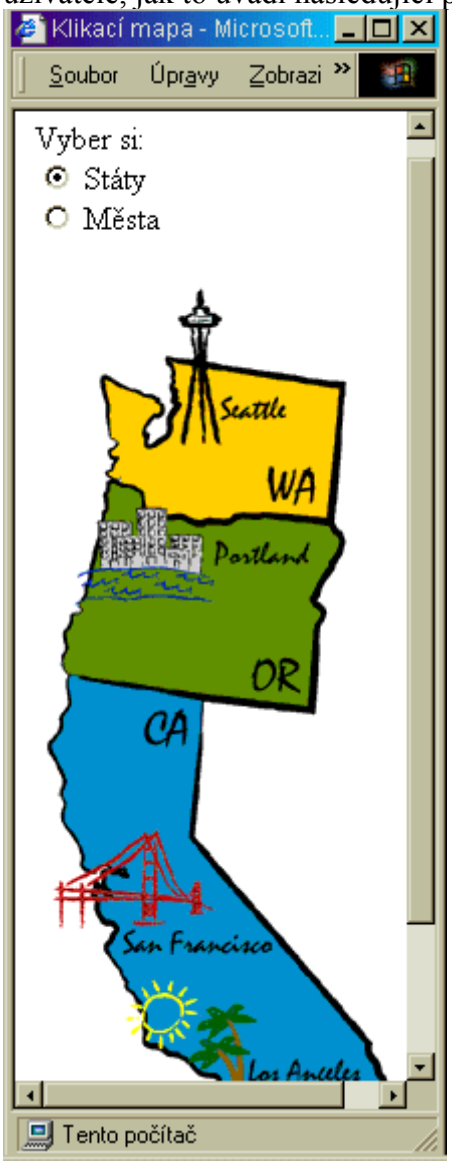
Jsou k dispozici dva typy map. Na straně uživatele, na straně klienta. Obrazové mapy na straně klienta využívají prvek *map* a jejich velikou výhodou je, že nevyžadují opakovaný přístup na server. Prvek *map* obsahuje soubor prvků *area*, které definují souřadnice pro každou oblast kliknutí.

Prvek *map* musí být pojmenován pak s ním může být spojen libovolný počet obrázků a to pomocí atributu obrázku *UseMap*. Níže uvedený příklad spojuje obrázek „mapa.gif“ s obrazovou mapou, pojmenovanou „mojemapa“.

```
<IMG SRC=“mapa.gif“ UseMap= “#mojemapa“>
```

Obrazovou mapu je možno umístit v dokumentu kamkoliv, bez ohledu na to zda je mapa s dokumentem spojena. Více obrázků může sdílet jedinou obrazovou mapu.

Velmi výhodnou aplikací dynamické změny obrazové mapy je zajištění různého stupně zrnitosti v zeměpisných mapách. V následujícím příkladě, je lepší ovladatelnost zajištěna tím, že uživatele necháme nejprve definovat podmnožinu položek, o které se zajímá. Tím, že uživateli umožníme aby se rozhodl mezi státy a městy, se výběr stává mnohem jednodušším. Toto filtrování lze provést přepínáním mezi dvěma mapami daného obrázku v závislosti na výběru uživatele, jak to uvádí následující příklad. (viz. obr. 3)



obr. 3

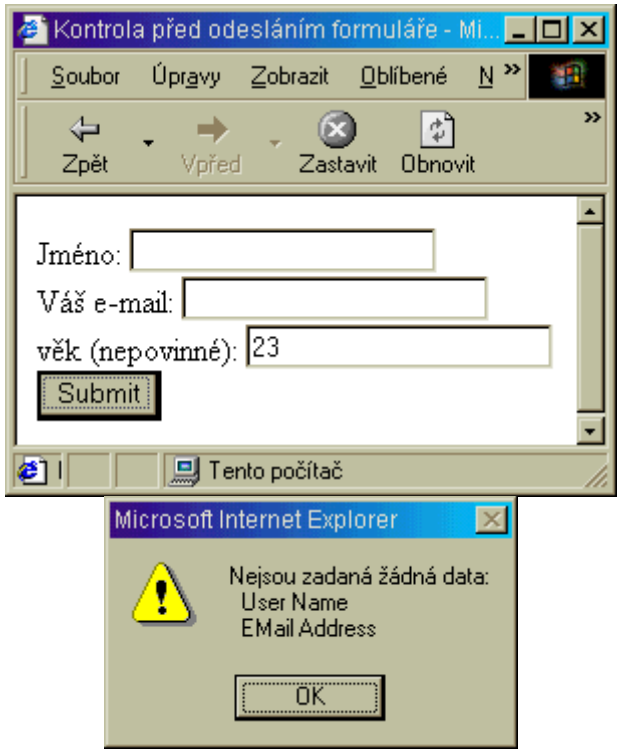
```
<HTML>
  <HEAD>
```

```
    <TITLE>Klikací mapa </TITLE>
    <SCRIPT LANGUAGE="JavaScript">
      function setMap(mapName) {
        document.all.mapImage.useMap = mapName;
      }
    </SCRIPT>
  </HEAD>
  <BODY>
    <P>Vyber si:<BR>
      <INPUT TYPE=RADIO NAME="feature" ID="States" Value="#States"
        ONCLICK="setMap(this.value);" CHECKED>
      <LABEL FOR="States">Státy</LABEL><BR>
      <INPUT TYPE=RADIO NAME="feature" ID="Cities" Value="#Cities"
        ONCLICK="setMap(this.value);">
      <LABEL FOR="Cities">Města</LABEL></P>
    <P><IMG ID="mapImage" SRC="places.gif" BORDER=0
      WIDTH=197 HEIGHT=448 USEMAP="#States"></P>
    <MAP NAME="Cities">
      <AREA SHAPE="POLYGON" HREF="la.htm"
        COORDS="108, 408, 164, 407, 165, 388, 111, 387, 109, 361, 86, 361,
                73, 394, 94, 411">
      <AREA SHAPE="POLYGON" HREF="sanfran.htm"
        COORDS="12, 301, 58, 275, 75, 305, 80, 301, 87, 314, 92, 326, 119,
                329, 121, 340, 45, 341, 44, 328, 9, 328">
      <AREA SHAPE="POLYGON" HREF="portland.htm"
        COORDS="34, 120, 47, 120, 49, 115, 68, 115, 69, 123, 86, 127, 86,
                131,140, 131, 137, 144, 86, 145, 91, 162, 22, 160, 22,
                148, 26, 144">
      <AREA SHAPE="POLYGON" HREF="seattle.htm"
        COORDS="73, 86, 93, 84, 92, 73, 125, 73, 123, 59, 92, 57, 87, 43, 93,
                22, 82, 2, 71, 21, 79, 45">
    </MAP>
    <MAP NAME="States">
      <AREA SHAPE="POLYGON" HREF="california.htm"
        COORDS="14, 204, 18, 200, 83, 209, 79, 278, 166, 386, 171, 403, 167,
409, 166, 419, 163, 423, 164, 430, 166, 436, 161, 439, 115, 438, 112, 433, 110,
420, 97, 409, 92, 401, 82, 399, 77, 392, 56, 385, 54, 369, 46, 357, 46, 352, 34,
338, 39, 327, 35, 322, 32, 309, 34, 297, 25, 297, 24, 288, 14, 273, 15, 255, 9,
235, 12, 224, 12, 221, 16, 216">
      <AREA SHAPE="POLYGON" HREF="oregon.htm"
        COORDS="16, 199, 136, 216, 140, 178, 143, 171, 138, 164, 153, 132,
147, 122, 103, 120, 80, 123, 72, 121, 55, 121, 51, 109, 37, 105, 22, 163, 23,
166, 18, 173, 14, 189">
      <AREA SHAPE="POLYGON" HREF="washington.htm"
```

```
        COORDS="33, 50, 64, 64, 57, 74, 57, 86, 63, 81, 70, 65, 66, 41, 152,
55, 147, 123, 100, 119, 86, 124, 74, 120, 56, 119, 51, 108, 40, 104, 36, 99, 43,
93, 37, 87, 41, 84, 36, 80">
    </MAP>
  </BODY>
</HTML>
```

3.4 Ověření formuláře při odeslání

Když chceme zjistit zda jsou zadané prvky ve formuláři platné, nebo potřebujeme zjistit jestli jsou zadány všechny požadované informace, použijeme kontrolu údajů v době odesílání. Níže uvedená ukázka demonstruje způsob rozšíření vnitřního textového rámečku pomocí atributu *required,* tak aby bylo zajištěno že je klient vyplnil. (viz. přílohy obr. 4)



obr. 4

```
<HTML>
  <HEAD>
    <TITLE>Kontrola před odesláním formuláře</TITLE>
    <SCRIPT LANGUAGE="JavaScript">
      function isEmpty(str) {
```

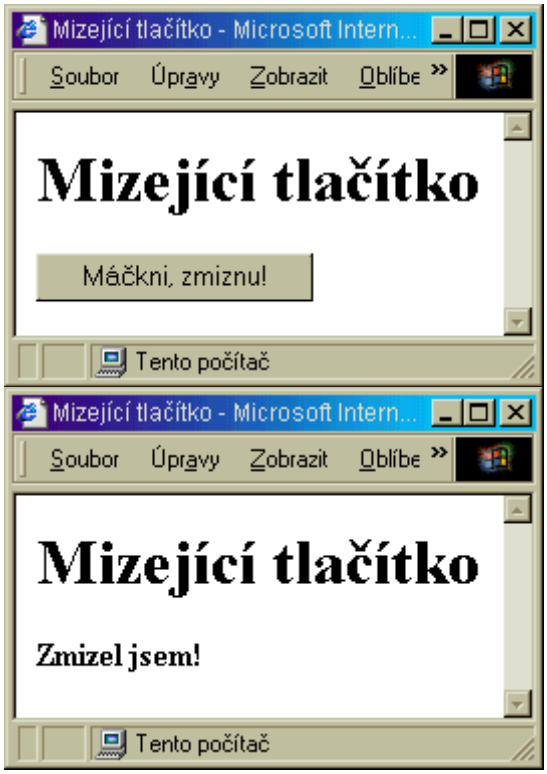
```
        for (var intLoop = 0; intLoop < str.length; intLoop++)
          if (" " != str.charAt(intLoop))
            return false;
        return true}
      function checkRequired(f) {
        var strError = "";
        for (var intLoop = 0; intLoop<f.elements.length; intLoop++)
          if (null!=f.elements[intLoop].getAttribute("required"))
            if (isEmpty(f.elements[intLoop].value))
              strError += "  " + f.elements[intLoop].name + "\n";
        if ("" != strError) {
          alert("Nejsou zadaná žádná data:\n" + strError);
          return false;
        }
      }
    </SCRIPT>
  </HEAD>
  <BODY>
    <FORM NAME="demo" ONSUBMIT="return checkRequired(this);">
      Jméno:
        <INPUT TYPE=TEXT NAME="User Name" required><BR>
      Váš e-mail:
        <INPUT TYPE=TEXT NAME="EMail Address" required><BR>
      věk (nepovinné):
        <INPUT TYPE=TEXT NAME="Age"><BR>
      <INPUT TYPE=SUBMIT VALUE="Submit">
    </FORM>
  </BODY>
</HTML>
```

3.5 Mizející tlačítko

Nyní si ukážeme efekt zmizení tlačítka. Je zde použito vlastnosti prvku *outerHTML,* který zpřístupňuje celý obsah prvku. (viz. přílohy obr. 5)

```
<HTML>
  <HEAD>
    <TITLE>Mizející tlačítko</TITLE>
  </HEAD>
  <BODY>
    <H1>Mizející tlačítko</H1>
    <INPUT TYPE=BUTTON VALUE="Máčkni, zmiznu!"
      ONCLICK="this.outerHTML = '<B>Zmizel jsem!</B>'">
```

```
  </BODY>
</HTML>
```



obr.5

Další vlastnosti obsahu:

mohou být buď :
a) HTML *innerHTML* a *outerHTML,*
b) textové *innerText* a *outerText*

- rozdíl spočívá v tom, že vlastnosti HTML se týkají vybraného prvku; vlastnosti textové představují pouze obsah prvku bez jeho značek. Ukážeme to na jednoduchém příkladě:

```
<H1>Hraje Vám<EM>Petra</EM>klavíristka.</H1>
```

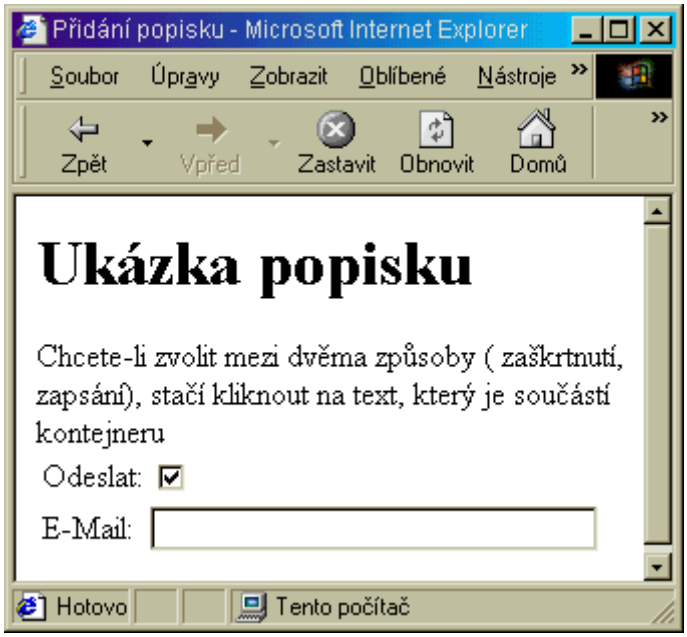
Potom:

innerHTML	Hraje Vám<EM> Petra</EM> klavíristka.
OuterHTML	<H1>Hraje Vám<EM>Petra</EM> klavíristka.</H1>
innerText	Hraje Vám Petra klavíristka.

outerText Hraje Vám Petra klavíristka.

3.6 Přidání popisku

Prvky **Label** jsou určeny pro ovládací prvky, které vytvářejí spojení k záložkám. Když uživatel klepne na popisek, zobrazí se odpovídající ovládací prvek a stane se aktivním. U přepínačů a zaškrtávacích políček platí, že klepnutím na popisek se rovněž klepne na odpovídající tlačítko a změní se jeho hodnota. <LABEL> odkazuje na odpovídající prvek ovládajícího prvku pomocí atributu For. Ten obsahuje jedinečný identifikátor (ID) ovládacího prvku na stránce. (viz. obr. 6)



obr. 6

```
<HTML>
  <HEAD>
    <TITLE>Přidání popisku</TITLE>
</HEAD>
  <BODY>
    <H1>Ukázka popisku</H1>
    <P> Chcete-li zvolit mezi dvěma způsoby ( zaškrtnutí, zapsání),
          stačí kliknout na text, který je součástí kontejneru</ P>
    <TABLE>
      <TR>
```

```
        <TD NOWRAP>
          <LABEL FOR="Info">Odeslat: </LABEL>
        </TD>
        <TD>
          <INPUT ID="Info" TYPE="Checkbox" VALUE="Information">
        </TD>
      </TR>
      <TR>
        <TD NOWRAP>
          <LABEL FOR="Email">E-Mail: </LABEL>
        </TD>
        <TD>
          <INPUT ID="Email" TYPE="Text" SIZE=30>
        </TD>
      </TR></TABLE>
  </BODY>
</HTML>
```

Prvek **LABEL** také zavádí možnost propojit ovládací prvek s obsahem. Atribut **ACCESSKEY**. Ten obsahuje jediný znak, který lze použít jako klávesovou zkratku pro odkaz k ovládacímu prvku. Po stisku klávesy ALT a stisknutím znaku vyvoláte klávesovou zkratku. Následující program vytváří popisek s přístupovou klávesou.

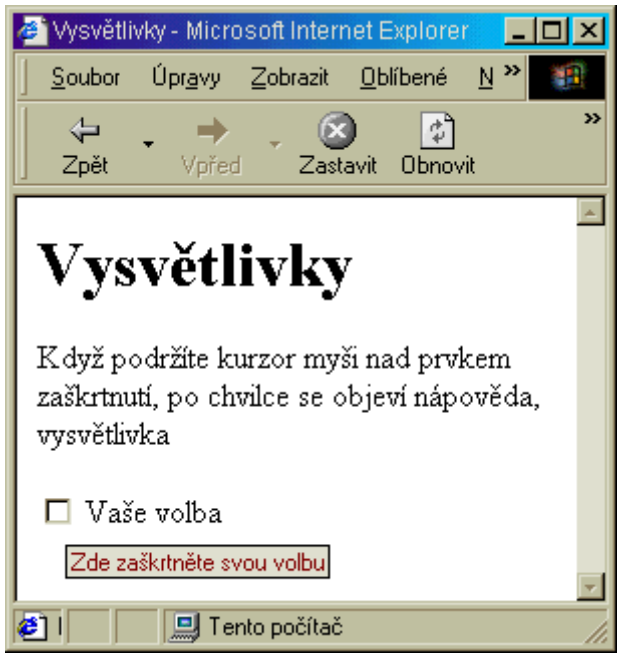
```
<“LABEL FOR=“txt“ ACCESSKEY=“v“>
<SPAN CLASS=“accesskey“> v </SPAN >Vítejte!
</LABEL>
< INPUT TYPE=TEXT ID=“txt“ SIZE=20 >
```

3.7 Přidání vysvětlivek

Jde o takzvaný ToolTip, což je malé okno s textem , které se zobrazí, když se ukazatel myši podrží na prvku. ToolTip může být spojen s jakýmkoliv ovládacím prvkem a nadpisem. V následujícím programu obsahuje zaškrtávací políčko atribut **TITLE**, jenž zobrazí ToolTip, když je nad ním podržen kurzor myši. (viz. obr. 7)

```
<HTML>
  <HEAD>
    <TITLE>Vysvětlivky </TITLE>
  </HEAD>
  <BODY>
    <H1>Vysvětlivky</H1>
```

```
   <P>Když podržíte kurzor myši nad prvkem zaškrtnutí, po chvilce se objeví
        nápověda, vysvětlivka</P>
   <INPUT ID="Info" TYPE="Checkbox" VALUE="Information"
     TITLE="Zde zaškrtněte svou volbu"> Vaše volba
  </BODY>
</HTML>
```



obr. 7

3.8 Button, novinky

Nový prvek **Button** umožňuje zobrazení bohatého obsahu formou tlačítka. Tlačítko lze nyní vytvořit tak, jak to umožňuje HTML a popis stylu. (Tedy velmi kreativně). Nevýhodou je že starší prohlížeče to nezobrazí a je nutno je definovat v <INPUT>. (viz. obr. 8)

```
<HTML>
  <HEAD>
    <TITLE>Nový Button </TITLE>
  </HEAD>
  <BODY>
    <H1>Button Element</H1>
    <P>Ukázka nového tlačítka. První má modrozelený nápis. Druhý je ve
žlutém
        políčku.
```

```
   <P><BUTTON STYLE="font-family: Arial; font-size:16pt; color:blue">
     Modrá barva<SPAN STYLE="font-style:italic;color:green">zelená!
                 </SPAN>
   </BUTTON>
   <P><BUTTON STYLE="background:URL(cool.gif)
                              yellow;font-weight:bold" >
    <P ALIGN="Center">Odeslat</P>
    <P ALIGN="Center" STYLE="font-style:italic">Nyní</P>
</BUTTON>
 </BODY>
</HTML>
```

obr. 8

3.9 Seskupení prvků

Jde o prvek **Fieldset**. Je určen pro seskupení množin. Například ovládacích prvků, pro jejich lepší přehlednost. Fieldset může obsahovat jednu legendu, jenž je zobrazena na okraji rámečku. Za ní pak mohou následovat libovolné prvky HTML. (viz. obr. 9)

```
<HTML>
 <HEAD>
  <TITLE>Prvek seskupení</TITLE>
 </HEAD>
 <BODY>
  <H1>Prvek seskupení</H1>
   <FIELDSET STYLE="width: 5em">
   <LEGEND>Ovoce </LEGEND>
   <INPUT TYPE="Radio" VALUE="jablko" NAME="SIZE" ID="jablko">
   <LABEL FOR="BIG">Jablko</LABEL><BR>
   <INPUT TYPE="Radio" VALUE="hruska" NAME="SIZE" ID="hruska">
   <LABEL FOR="SMALL">Hruška</LABEL>
  </FIELDSET>
 </BODY>
</HTML>
```

obr. 9

3.10 Marquee je tu!

Marquee. Umí zobrazovat text HTML. A roluje text jak doleva a doprava tak má schopnost rolovat text i nahoru a dolů. Prvek Marqueee umí zobrazit také tabulky a různé ovladače událostí. Navíc je prvek nyní představován objektem v objektovém modelu. Níže uvedená ukázka uvádí rolování směrem nahoru (viz. přílohy obr. 10) :

```
<HTML>
```

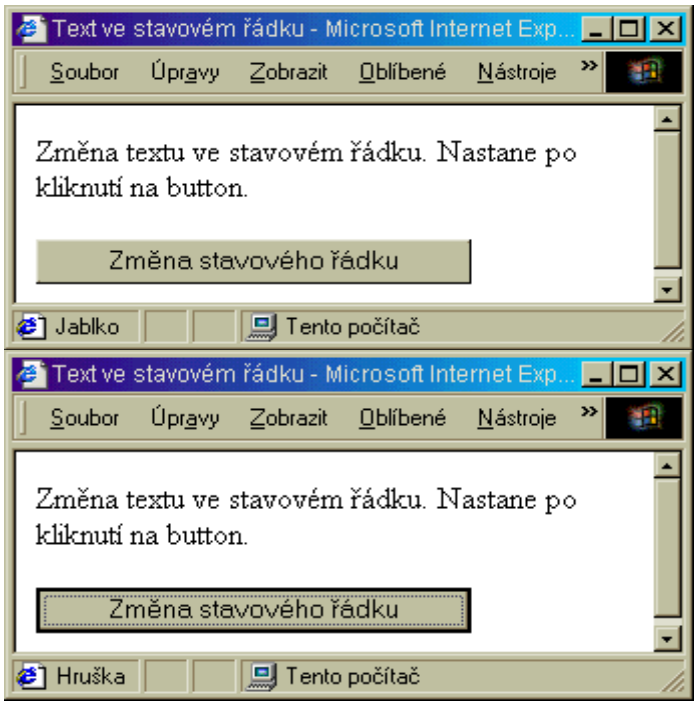
```
<HEAD>
  <TITLE>Objekt Marquee</TITLE>
</HEAD>
<BODY>
  <H1>Rolování textu pomocí Marquee</H1>
  <MARQUEE STYLE="height: 150px; border: 1pt navy solid; width: 100pt"
                  DIRECTION="Up">
   <TABLE align=center>
     <CAPTION>Tohle je ono!</CAPTION>
     <TR><TD>řádek první</TD><TD>všechny</TD></TR>
     <TR><TD>řádek druhý</TD><TD>řádky</TD></TR>
     <TR><TD>řádek třetí</TD><TD>se nám</TD></TR>
     <TR><TD>řádek čtvrtý</TD><TD>pěkně</TD></TR>
     <TR><TD>řádek pátý</TD><TD>rolují</TD></TR>
     <TR><TD>řádek šestý/TD><TD>nahoru</TD></TR>
   </TABLE>
 </MARQUEE>
</BODY>
</HTML>
```

obr. 10

3.11 Stavový řádek

Přístup k zprávám stavového řádku je umožněn přes dvě vlastnosti. Jsou to *defaultStatus* a *status*. Jsou to řetězce určené pro čtení a zápis.

status	používá se pro zobrazení dočasných hlášení
defaultStatus	používá se k zobrazení hlášení, které trvá tak dlouho dokud není změna nebo dokud není opuštěno okno prohlížeče (viz. obr. 11)



obr. 11

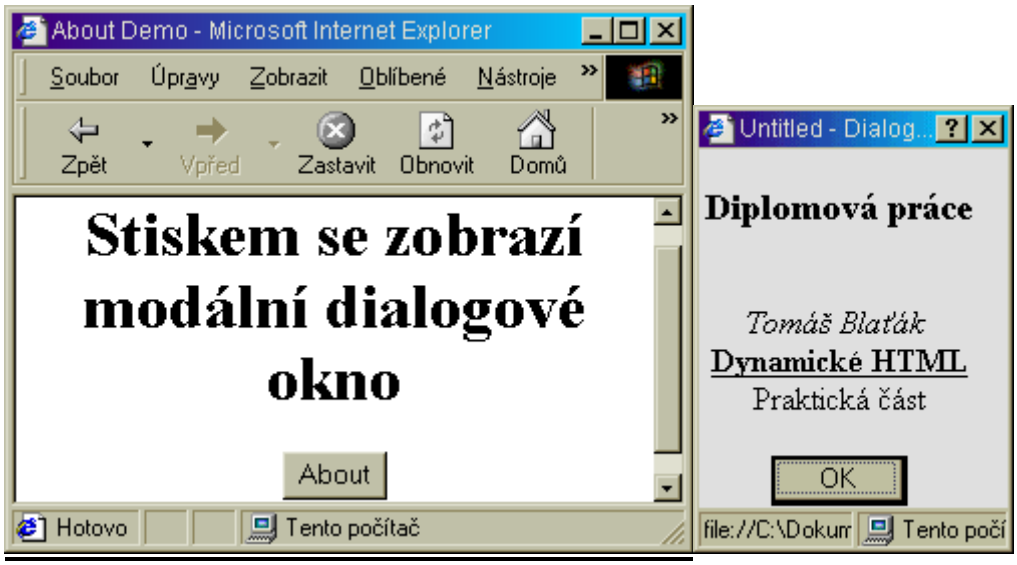
```
<HTML>
  <HEAD>
    <TITLE>Text ve stavovém řádku</TITLE>
    <SCRIPT LANGUAGE="JavaScript">
      function setStatus() {
        window.defaultStatus = "Jablko";
        window.status = "Hruška";
      }
    </SCRIPT>
  </HEAD>
  <BODY>
```

```
    <P>Po klepnutí se zobrazí  ve stavovém řádku Hruška, po odjetí myší se
        vrátí Jablko < P>
     <FORM>
       <INPUT TYPE=BUTTON VALUE="Změna stavového řádku"
        ONCLICK="setStatus();">
     </FORM>
  </BODY>
</HTML>
```

3.12 Modální okna

Metodou *showModaldialog se* provádí vytvoření vlastních dialogových oken, která mohou zobrazovat HTML soubory. Uvnitř dialogového okna je model částečně odlišný od klasického modelu okna, protože dialogové okno není úplnou instancí prohlížeče, ale spíše zobrazovačem HTML. Modální okna se liší:

- Nedochází k navigaci okna. Kliknutí na propojení otevře novou adresu v nové
  instanci prohlížeče.
- Obsah dialogového okna je neoznačitelný.



obr.12

```
<HTML>
  <HEAD>
    <TITLE>About Demo</TITLE>
```

```
    <SCRIPT LANGUAGE="JavaScript">
      function about() {
        event.srcElement.blur();
        window.showModalDialog("about.html","","dialogWidth:10em;
        dialogHeight:14em")
      }
    </SCRIPT>
  </HEAD>
 <BODY>
   <CENTER>  <H1>Stiskem se zobrazí modální dialogové okno</H1>
    <INPUT TYPE=BUTTON VALUE="About"ONCLICK="about();">
   </CENTER>
</BODY>
</HTML>
```

Modální dialogová okna jsou určena pro zobrazování zpráv, které vyžadují odpověď. Při zobrazení okna musí být vždy doplněno i zavírací tlačítko. Jinak je okno řádně uzavřitelné jen „křížkem“. Příklad uveden níže. (viz. obr. 12)

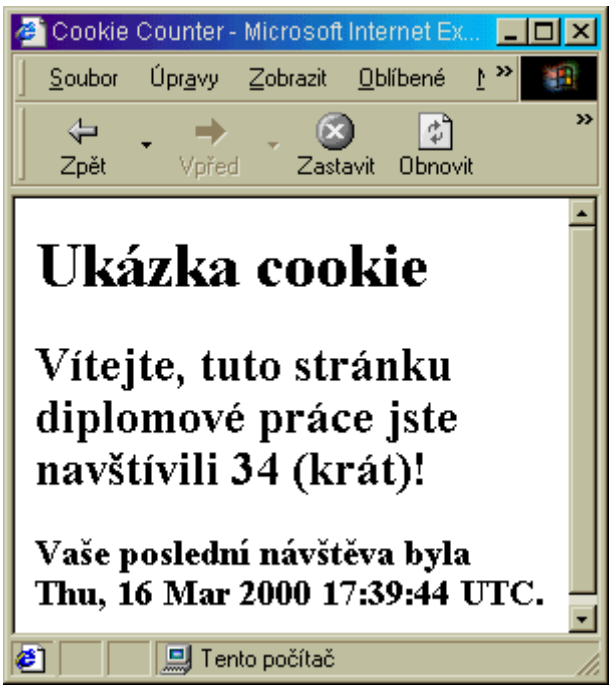
```
<HTML>
 <HEAD>
     <TITLE>Untitled</TITLE>
 </HEAD>
 <BODY bgcolor=#E6E6E6>
  <CENTER><BR>
  <H3>Diplomová práce</H3><BR>
  <I>Tomáš Blaťák</I><BR>
  <U><B>Dynamické HTML</B></U><BR>
  Praktická část<BR><BR>
  <INPUT TYPE=SUBMIT VALUE="OK"  STYLE="width:5em"
  ONCLICK="window.close( );">
  <BR>
  </CENTER>
  </BODY>
</HTML>
```

3.13 Cookie

Následující program demonstruje způsob používání cookie pro počítání počtu návštěv a zároveň zobrazuje datum polední návštěvy stránky. Program využívá funkci *parseCookie* a *setCookie.* (viz. obr. 13)

```
<HTML>
  <HEAD>
    <TITLE>Cookie Counter</TITLE>
  </HEAD>
  <BODY>
    <H1>Ukázka cookie</H1>
    <SCRIPT LANGUAGE="JavaScript">
      function parseCookie() {
       // Oddělení každého cookie
       var cookieList = document.cookie.split("; ");
       //Pole pro každé cookie v seznamu cookieList
       var cookieArray = new Array();
       for (var i = 0; i < cookieList.length; i++) {
          // Oddělení  dvojic name-value.
          var name = cookieList[i].split("=");
          //Dekódování a přidání k poli cookie.
          cookieArray[unescape(name[0])] = unescape(name[1]);
       }
       return cookieArray;
      }
      // Tento program vyžaduje funkci parseCookie.
      function setCookie(visits) {
       /*Tato rutina nastavuje cookie nastavením jeho hodnoty na počet návštěv
          nastavením jeho doby platnosti na jeden rok od chvíle. */
       var expireDate = new Date();
       var today = new Date();
       //Nastavení doby platnosti do budoucna.
       expireDate.setDate(365 + expireDate.getDate());
       //Uložení počtu návštěv.
       document.cookie = "Visits=" + visits + "; expires=" +
         expireDate.toGMTString() + ";";
       // Uložení dnešního data a času jako poslední návštěvy.
       document.cookie = "LastVisit=" +
            escape(today.toGMTString()) +
            "; expires=" + expireDate.toGMTString() + ";";
       }
       if ("" == document.cookie) {
         // Spouští cookie.
         setCookie(1);
         document.write("<H2>This is your first visit to our " +
           "humble home page.</H2>");
       }
       else {
```

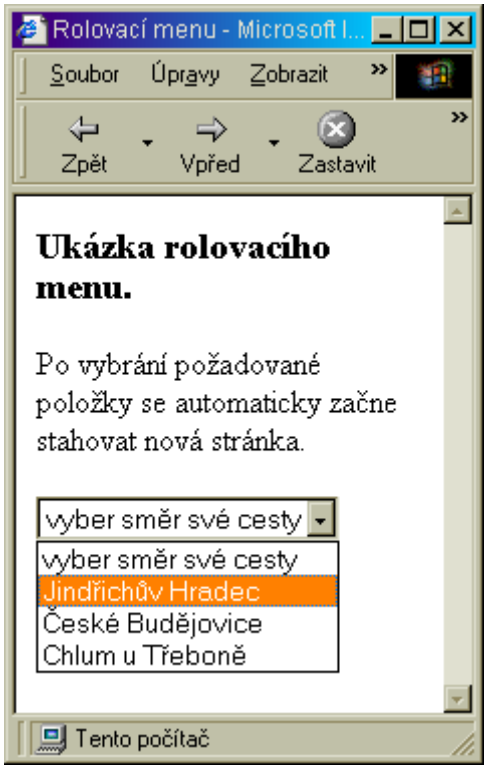
```
        // Rozdělení cookie.
        var cookies = parseCookie();
        // Vyšle zprávu Welcome Back a zvýší počet návštěv.
        document.write("<H2>Vítejte tuto stránku jste navštívili po " +
          cookies.Visits++ + " (té)!</H2>");
        document.write("<H3>Vaše poslední  návštěva byla" +
          cookies.LastVisit + ".</H3>");
        // Nahradí současné cookie novým.
        setCookie(cookies.Visits);
      }
    </SCRIPT>
  </BODY>
</HTML>
```



obr. 13

3.14 Rolovací menu

Příklad tohoto nabídkového menu se liší od běžných nabídek tím, že volená položka se začne načítat ihned po jejím vybrání. Není třeba nejprve položku vybrat a potom teprve potvrdit její načtení kliknutím na odkaz. Tento příklad nám nabízí výběr mezi jízdními řády do třech směrů. Program je proveden tak, že každé proměnné je přiřazen index, kterému přísluší daná URL. Ve vlastním těle dokumentu je použito klasické Select menu. (viz. obr. 14)



obr. 14

```
<HTML>
<HEAD>
        <TITLE>Rolovací menu</TITLE>
<SCRIPT LANGUAGE = "JavaScript">
<!--
function MakeArray()
      {
      this.length = MakeArray.arguments.length
      for (var i = 0; i < this.length; i++)
      this[i+1] = MakeArray.arguments[i]
      }
var url = new MakeArray("",  "rady1.html", "rady2.html", "rady3.html");
function jumpPage(form) {
      i = form.SelectMenu.selectedIndex;
      if (i == 0) return;
      window.location.href = url[i+1];
} // -->
</SCRIPT>
</HEAD>
```

```
 <BODY>
<H3>Ukázka rolovacího menu.</H3>
  Po vybrání požadované položky se automaticky začne stahovat nová stránka.
   <FORM >
            <SELECT NAME="SelectMenu"
                     onChange="jumpPage(this.form)">
            <OPTION>vyber směr své cesty
            <OPTION>Jindřichův Hradec
            <OPTION>České Budějovice
            <OPTION>Chlum u Třeboně</SELECT>
   </FORM> </BODY>
</HTML>
```

3.15 Navigační panel založený na Link

Prvku Link může být také využito pro definování vztahů mezi různými typy dokumentů. V našem příkladě se zaměříme na definování vztahů mezi dokumenty a to pomocí atributů REL a HREF, k nimž lze přistupovat pomocí skriptů. Ukážeme si vytvoření navigačního panelu, který načítá každý prvek Link dokumentu, aby zjistil následné a předchozí dokumenty. Kdykoliv je dokument zaveden volá funkci v dané množině rámců, aby podle nových odkazů dokumentu aktualizovala navigační tlačítka. (viz. obr. 15)

Dokument Links

Dokument links.html (který je uveden níže) definuje množinu rámců a obsahuje jádro programu pro správu mezi odkazy na stránce a navigačním panelem. Každý zobrazený dokument musí volat funkci *setupLinks,* aby byla provedena aktualizace navigačního panelu v navigační části okna. Když se stránka odstraňuje z paměti , je volána funkce *clearLinks*, aby byla všechna tlačítka vztahů znepřístupněna a tím se zajistí, že odkaz bude fungovat, když klient přejde na stránku, pro kterou nejsou definovány.

```
<HTML>
 <HEAD>
   <TITLE>Vztahy mezi odkazy</TITLE>
   <SCRIPT LANGUAGE="JavaScript">
     function setButton(b, dis, title, href) {
       b.disabled = dis
       b.title = title
       b.href = href
     }
     function updateBar(doc) {
```

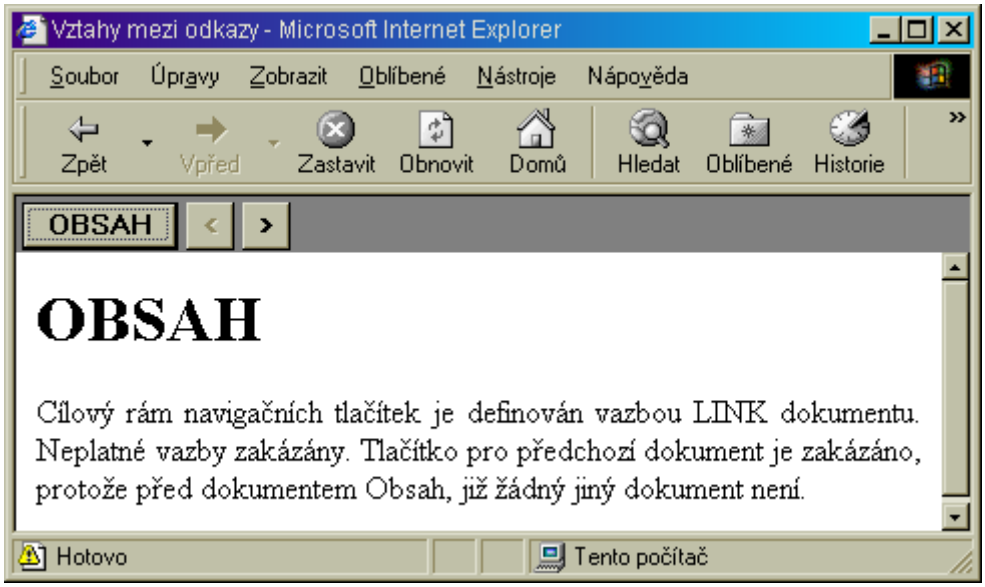
```
      var links = doc.all.tags("LINK");
      var navDoc = window.navigation.document.all;
         clearLinks();
      for (var intLink=0; intLink < links.length; intLink++) {
       var el = links[intLink];
       if ("previous"==el.rel) {
        setButton(navDoc.previous, false, el.title, el.href)
       };
       if ("next"==el.rel) {

        setButton(navDoc.next, false, el.title, el.href)
       };
      };
     };
     function clearLinks() {
      var navDoc = window.navigation.document.all;
      // Initialize buttons by disabling them and removing the title.
      with (navDoc) {
       setButton(previous, true, "", "")
       setButton(next, true, "", "")
      };
     };
   </SCRIPT>
  </HEAD>
  <FRAMESET ROWS="28,*" BORDER=0>
   <FRAME SRC="navigate.htm" NAME="navigation" SCROLLING=NO>
   <FRAME SRC="contents.htm" NAME="contents">
  </FRAMESET>
</HTML>
```

Dokument Contents

Součástí aplikace je také dokument contents.htm, který definuje vztah odkazu k dalšímu dokumentu v posloupnosti. Při zavádění volá dokument funkci *stupLinks*, aby aktualizoval dostupné odkazy, a při odstraňování dokumentu musí volat funkci *clearLinks*. Zdrojový kód je uveden níže.

```
<HTML>
  <HEAD>
   <TITLE>Obsah</TITLE>
  <LINK rel="next" href="chapter1.htm" title="Chapter 1">
  </HEAD>
  <BODY ONLOAD="parent.updateBar(window.document)"
```

```
            ONUNLOAD="parent.clearLinks();">
  <H1>OBSAH</H1>
  <P align=justify>Cílový rám navigačních tlačítek je definován vazbou LINK
     dokumentu. Neplatné vazby zakázány. Tlačítko pro předchozí dokument
     je zakázáno, protože před dokumentem Obsah, již žádný jiný dokument
     není.
 </BODY>
</HTML>
```



obr. 15

Poslední součástí této aplikace je dokument, který vytváří samotný navigační panel. Je zde použito klasických tlačítek Button. Zdrojový text uveden níže.

```
<HTML>
 <HEAD>
  <TITLE>Navigační panel</TITLE>
  <STYLE TYPE="text/css">
    body {margin-top: 2pt; margin-left: 2pt; background: gray}
    input {font-weight: bold}
  </STYLE>
 </HEAD>
 <BODY>
  <INPUT TYPE="Button" VALUE="TOC" TITLE="Table of Contents"
            ONCLICK="parent.contents.location='contents.htm'">
  <INPUT TYPE="Button" ID="previous"  VALUE=" < "
```

```
            ONCLICK="parent.contents.location=this.href">
   <INPUT TYPE="Button" ID="next" VALUE=" > "
            ONCLICK="parent.contents.location=this.href">
  </BODY>
</HTML>
```

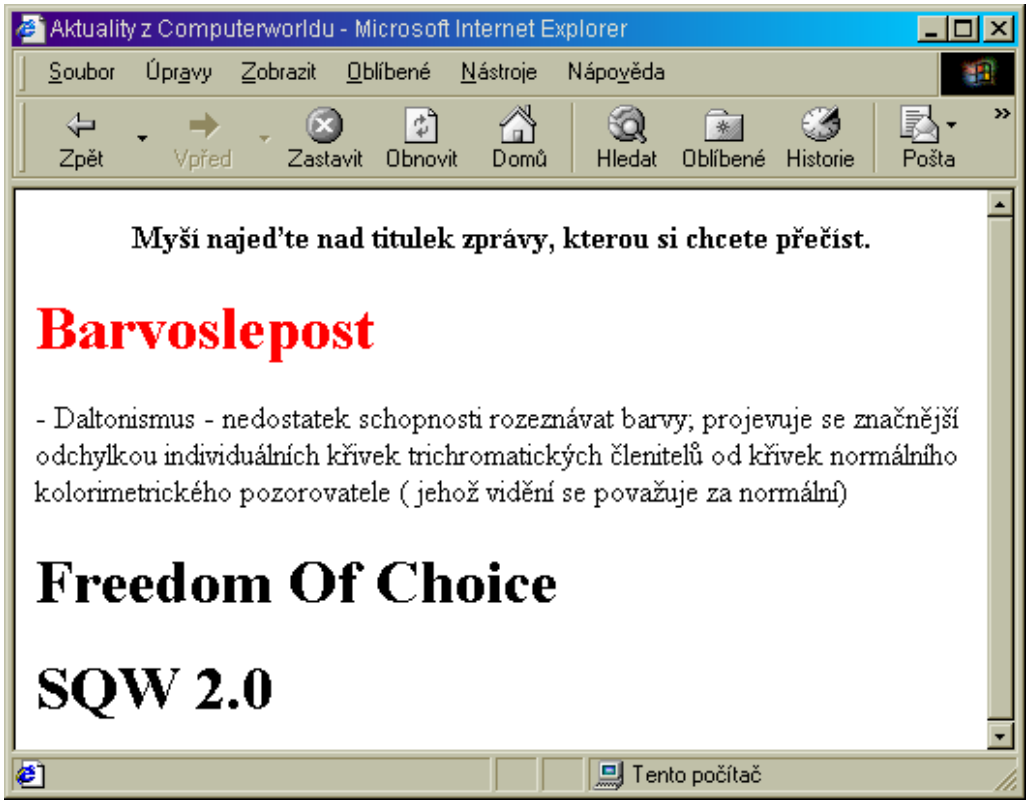
3.16 Zobrazování dokumentu na požadavek uživatele

Tato ukázka je pro předvedení možností dynamického HTML přímo typická. Nahraje se stránka, které obsahuje několik zpráv, ale zobrazí se pouze jejich nadpisy. Samotný text zpráv bude skryt nastavením vlastnosti *display* na hodnotu *none*. Celá zpráva se zobrazí až po najetí myší na jeho název. Navíc se nadpis zobrazené zprávy barevně zvýrazní. Podobný mechanismus můžeme použít i pro několikaúrovňové seznamy, které umožníme postupně rozbalovat do nižších úrovní. Všimněme si, že jednotlivé zprávy jsou obsaženy v elementu `DIV`, aby pro ně šlo nastavit společné reagování na události. Na ukázce je rovněž vidět, jaké úspory přináší používání funkcí a jejich parametrizování. Definovali jsme dvě funkce -- *Display*() a *Undisplay*(). Obě jako parametr očekávají číslo zprávy. *Display* `()` zobrazí obsah zprávy a zvýrazní její nadpis. Nadpisy musí mít rovněž přiřazeny správné ID. *Undisplay* `()` se volá po opuštění elementu myší a text zprávy schová a barvu nadpisu nastaví zpět na černou. (viz. obr. 16)

```
<HTML>
<HEAD>
<TITLE>Aktuality z Computerworldu</TITLE>
<SCRIPT LANGUAGE="JavaScript">
<!--
function Display(elementID)
{
var id;

   id = "Zprava" + elementID;
   document.all.item(id).style.display = "";
   id = "Nadpis" + elementID;
   document.all.item(id).style.color = "red";
}

function Undisplay(elementID)
{
var id;

   id = "Zprava" + elementID;
```

```
   document.all.item(id).style.display = "none";
   id = "Nadpis" + elementID;
   document.all.item(id).style.color = "black";
}
// -->
</SCRIPT>
</HEAD>
<BODY>

<DIV ALIGN=CENTER>
<B>Myší najeďte nad titulek zprávy, kterou si chcete přečíst.</B>
</DIV>

<DIV onMouseOver="Display('1')" onMouseOut="Undisplay('1')">
<H1 ID=Nadpis1>Barvoslepost</H1>
<DIV ID=Zprava1 STYLE="display: none">
<P>- Daltonismus - nedostatek schopnosti rozeznávat barvy; projevuje se
značnější odchylkou individuálních křivek trichromatických členitelů od
křivek normálního kolorimetrického pozorovatele ( jehož vidění se považuje za
normální)
</DIV>
</DIV>

<DIV onMouseOver="Display('2')" onMouseOut="Undisplay('2')">
<H1 ID=Nadpis2>Freedom Of Choice</H1>
<DIV ID=Zprava2 STYLE="display: none">
<P>Společnost Netscape oznámila záměr změnit stav, kdy podíl jejího
browseru na trhu klesá. Po rozhodnutí soudu o nepřípustnosti vázat
instalaci Windows 95 na Internet Explorer, zahajuje Netscape především
na Internetu mohutnou reklamní kampaň pojmenovanou "Freedom Of Choice".
Představitelé společnosti údajně zvažují i možnost, že např. separovaný
Navigator by mohl být šířen zdarma.
</DIV>
</DIV>

<DIV onMouseOver="Display('3')" onMouseOut="Undisplay('3')">
<H1 ID=Nadpis3>SQW 2.0</H1>
<DIV ID=Zprava3 STYLE="display: none">
<P>Společnost Corpus oznámila verzi 2.0 svého vývojového prostředí SQW.
Největší zlepšení oproti předešlé verzi by mělo spočívat ve zvýšení
rychlosti provozu dynamicky generovaných databázových aplikací.
</DIV>
</DIV>
```

```
</BODY>
</HTML>
```



obr. 16

3.17 Plynulá změna pomocí filtrů

Jednou z možností, kterou při přechodu z jednoho elementu v druhý nabízejí filtry, jsou přechodové filtry. Přechodový filtr se jmenuje *revealTrans* a pracuje se s ním poněkud odlišně než s ostatními filtry. Filtr má dva parametry -- typ přechodu (*transition*) a dobu přechodu (*duration*). U elementu, který chceme změnit nejprve nastavíme typ přechodu a dobu přechodu. Poté zavoláme metodu *Apply* () filtru. Ta připraví vše potřebné pro provedení přechodu. Nyní pomocí vlastnosti *innerText* (nebo *src* u obrázků) přiřadíme elementu nový obsah. Přechod pak spustíme vyvoláním metody *Play* (). Ta spustí přechod a použije přitom zadaný typ přechodu. IE 4.0 podporuje 23 přechodů (typy 0--22). Přechod číslo 23 je náhodný a vybere jeden z předchozích typů.
Přechod můžeme kdykoliv okamžitě ukončit pomocí metody *Stop* (). Pokud chceme nějakou činnost navázat na skončení přechodu, využijeme toho, že po

skončení přechodu je vyvolána událost *onFilterChange*. Nyní vytvoříme stránku, která bude obsahovat čítač, který bude postupně zobrazovat čísla od 20 do 0. Při změně čísla se použije vždy nový náhodně vybraný způsob přechodu.
(viz. obr. 17)

```
<HTML>
<HEAD>
<TITLE>Ukázka přechodových filtrů</TITLE>
<SCRIPT LANGUAGE="JavaScript">
<!--
var i=20;
function Tick()
{
   i--;
   if (i<0) {         // ukončení odpočítávání
      window.event.srcElement.onFilterChange = "";
      return;
      }
   document.all.Counter.filters.revealTrans.transition =
      Math.round(Math.random()*23);   // náhodný výběr přechodu
   document.all.Counter.filters.revealTrans.Apply();
   document.all.Counter.innerText = i;
   document.all.Counter.filters.revealTrans.Play();
}
// -->
</SCRIPT>
</HEAD>
<BODY onLoad="Tick()">
<DIV ALIGN=CENTER>
<DIV ID=Counter
    STYLE="height: 150px; width: 200px; text-align: center;
          background-color: blue; color: yellow;
          font: bold 140px Arial, sans-serif;
          filter: revealTrans(duration=1)"
    onFilterChange="Tick()"> 20
</DIV>
</DIV>
</BODY>
</HTML>
```

obr. 17

3.18 Animace na stránce

Abychom na stránku mohli umístit nějakou animaci -- tj. element, který se pohybuje -- musíme v pravidelných intervalech měnit jeho pozici. V tom nám

výborně pomůže metoda *window.setInterval("*funkce*",* ms*)*. Funkce zadaná prvním parametrem se vyvolá každých *ms* milisekund.
My vytvoříme stránku, na které se bude od okrajů obrazovky odrážet obrázek DHTML. O pohyb obrázku a jeho odrážení od okrajů se stará funkce *Move* ( ) , která je volaná v pravidelných intervalech. Funkce skriptu by měla být zřejmá z komentářů. U obrázku jsme vlastnost *z-index* nastavili na -1, aby se obrázek pohyboval pod textem stránky. Stránky, které jsou moc živé, mohou některým uživatelům vadit, protože je ruší při čtení. Měli bychom proto používat krátké animace nebo nabízet viditelný způsob, jak animace ukončit.
(viz. obr. 18)

```
<HTML>
<HEAD><TITLE>Dynamické HTML -- odrážející se logo</TITLE>
<SCRIPT LANGUAGE="JavaScript">
<!--
var id,            // pomocná proměnná pro časovač
   stepX, stepY;         // krok v X a Y-směru

function Start()          // spuštění pohybu
{
   // umístění obrázku doprostřed obrazovky
   document.all.Logo.style.pixelLeft =
      document.body.offsetWidth / 2;
   document.all.Logo.style.pixelTop =
      document.body.offsetHeight / 2;

   // obrázek uděláme viditelný
   document.all.Logo.style.visibility = "visible";

   // náhodná inicializace směru a rychlosti pohybu
   stepX = (Math.random()+5) * 2 - 5;
   stepY = (Math.random()+5) * 2 - 5;

   // nastavení časovače
   id = window.setInterval("Move()",50);
}

function Stop()                  // ukončení pohybu
{
   // vypnutí časovače
   window.clearInterval(id);

   // "schování obrázku"
```

```
   document.all.Logo.style.visibility = "hidden";
}

function Move()                  // posun loga
{
   // posunutí obrázku
   document.all.Logo.style.pixelLeft += stepX;
   document.all.Logo.style.pixelTop += stepY;

   // odražení od levého okraje
   if (document.all.Logo.style.pixelLeft <= 0) stepX = -stepX;

   // odražení od pravého okraje
   if ( document.all.Logo.style.pixelLeft >=
       (document.body.offsetWidth - document.all.Logo.width
        - stepX - 22) ) stepX = -stepX;
             // 22 je magické číslo šířky scroll-baru

   // odražení od horního okraje
   if (document.all.Logo.style.pixelTop <= 0) stepY = -stepY;

   // odražení od dolního okraje
   if ( document.all.Logo.style.pixelTop >=
       (document.body.offsetHeight - document.all.Logo.height
        - stepY) ) stepY = -stepY;
}
// -->
</SCRIPT>
</HEAD>
<BODY onload="Start()">
<IMG ID="Logo"
    STYLE="visibility: hidden; position: absolute; z-index:-1"
    SRC="letim.gif" onClick="Stop()">
<H1>A přece létá...</H1>
Animaci můžete zastavit kliknutím na DHTML.
</BODY>
</HTML>
```

obr. 18

3.19 Světelné zdroje

Velice pěkné efekty lze dosáhnout použitím filtru *Light*. Ten umožňuje nad jeden element umístit až deset světelných zdrojů. Zdroje světla různým způsobem nasvěcují element a určují tak, která jeho část bude viditelná.
IE 4.0 podporuje tři druhy zdrojů světla: rovnoměrný (Ambient), bodový (Point) a kuželový (Cone). Zdroje světla nelze definovat přímo jako parametry filtru, musí se přidat až přímo ve skriptu voláním metody *addAmbient*, *addCone* nebo *addPoint*. Nejjednoduší je přidání rovnoměrného zdroje světla:

*addAmbient(*R, G, B, intenzita*)*

Parametry *R*, *G* a *B* určují intenzitu základních složek barvy světla. Mohou nabývat hodnot od 0 do 255. Celková intenzita světelného zdroje je určena pomocí posledního celočíselného parametru.
K přidání kuželového zdroje světla potřebujeme zadat více parametrů:

*addCone(*x1, y1, z1, x2, x2, R, G, B, intenzita, úhel*)*

Souřadnice (x1, y1, z1) udávají prostorové souřadnice zdroje světla, který svítí do bodu (x2, y2, 0). *úhel* určuje vrcholový úhel vrhnutého kuželu světla. Ostatní parametry mají stejný význam jako u rovnoměrného zdroje světla.
Přidání bodového zdroje světla je velmi jednoduché:

*addPoint(*x, y, z, R, G, B, intenzita*)*

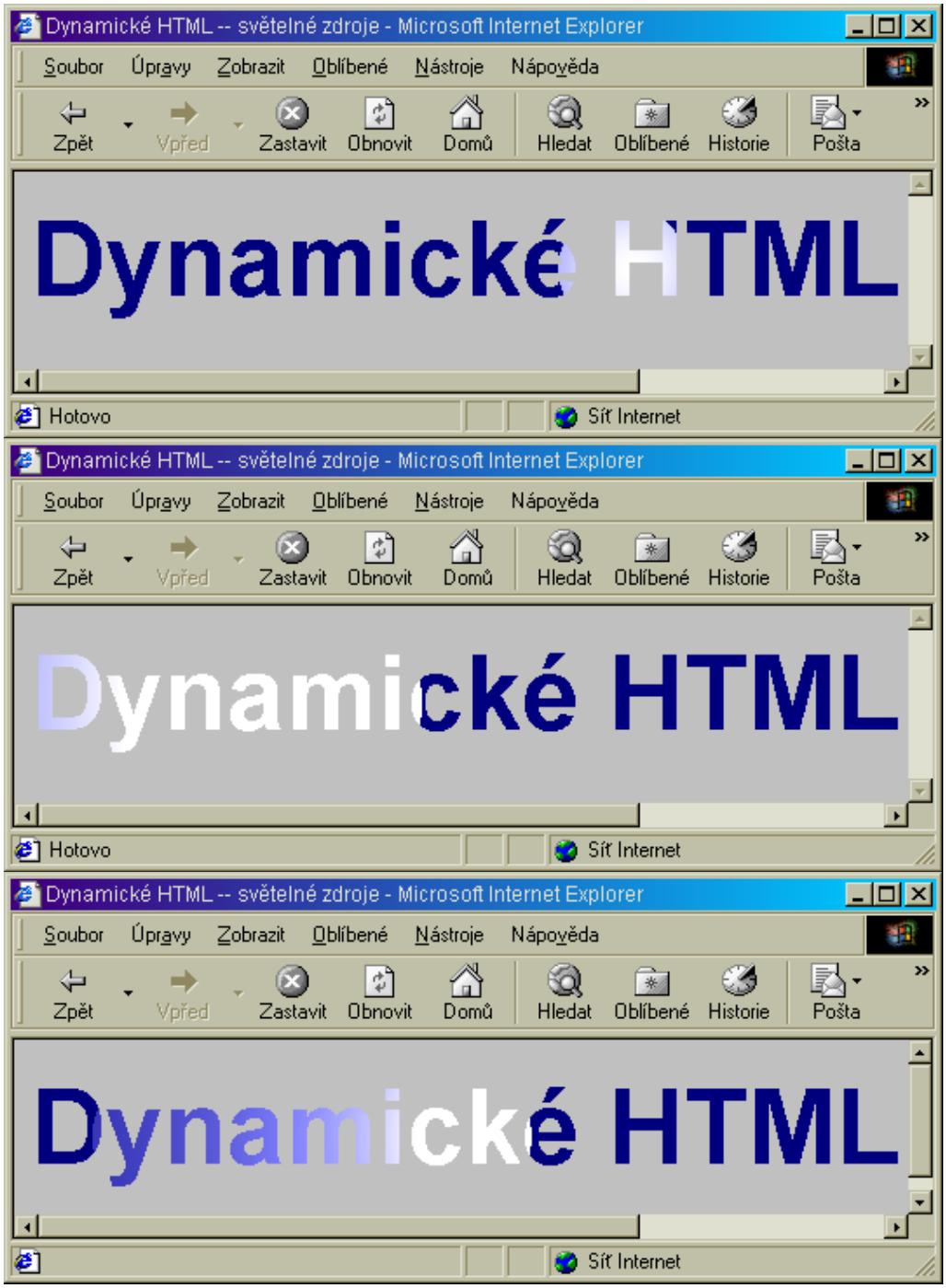
Po přidání světel můžeme měnit jejich polohu pomocí metody *moveLight* a vytvořit tak vskutku působivé efekty:

*moveLight(*číslo_světla, x, y, z, *true)*

Světla v jednom filtru jsou číslována od nuly podle pořadí v jakém byla přidána.
Praktické použití zdrojů světla si ukážeme na malé ukázce. Vytvoříme stránku s nápisem. Nápis bude částečně viditelný (díky zdroji rovnoměrného světla) a bude přes něj jezdit kužel světla, který vždy zviditelní osvětlenou část nápisu. (viz. obr.19)

```
<HTML>
 <HEAD>
    <TITLE>Dynamické HTML -- světelné zdroje</TITLE>
<STYLE type=text/css>BODY {
        BACKGROUND-COLOR: silver; COLOR: white;
        TEXT-ALIGN: center
}
#napis {
```

```
        FILTER: Light(); FONT: bold 60px Arial; HEIGHT: 80px; POSITION:
relative; WIDTH: 700px
}
</STYLE>
<SCRIPT language=JavaScript>
<!--
var MinX=0,
   MaxX=700,
   X=0,
   Y=40,
   Step=10,
   id;
function Move()
{
   // Přesun zdroje světla
   document.all.napis.filters.Light.moveLight(0, X, Y, 0, true);
   // Aktualizace
   X += Step;
   if (X >= MaxX) {
      // X = MaxX;
      Step = -Step;
      };
   if (X <= MinX) {
      // X = MinX;
      Step = -Step;
      };
}
function Init()
{
   document.all.napis.filters.Light.addCone((MinX+MaxX)/2, Y, 100, X, Y,
      255, 255, 255, 100, 20);    // Kuželový zdroj světla
   document.all.napis.filters.Light.addAmbient(0, 0, 127, 100);
                              // Rovnoměrný zdroj světla
   id = setInterval("Move()", 100);    // Aktivace pohybu
}// -->
</SCRIPT>
</HEAD>
<BODY onload=Init()>
<DIV id=napis>Dynamické HTML</DIV>
</BODY>
</HTML>
```



obr. 19

4.Závěr

Tato diplomová práce si nebrala za cíl podrobný popis jazyka DHTML. Jeho rozsáhlost a neustálý vývoj kupředu to neumožňují. Byl zde probrán základní objektový model a dále mnoho typických příkladů použití DHTML. Po jejich podrobném prostudování, by měl být uživatel schopen jejich začlenění do celého dokumentu, přestože všechny ukázky jsou schopny fungovat samostatně. Jde o moderní ukázky dnes běžně využívané na internetových stránkách. Je na každém, jakou aplikaci použije na své internetové stránce. Znovu bych si dovolil
zdůraznit, že ukázky jsou vytvořeny pro Internet Explorer 5.0 a jsou ke shlédnutí na přiložené disketě.

5. Seznam odborné literatury

[1] Broža Petr : Tvorba dokonalých WWW stránek pro úplné začátečníky; Computer press 1999; Brno

[2] Hlavenka J. a kol. : Vytváříme WWW stránky a spravujeme moderní Website; Computer press 1997; Praha

[3] Isaacs Scott : Dynamické HTML, druhé vydání; Computer press 2000; Praha

[4] Kosek Jiří : HTML, tvorba dokonalých www stránek, podrobný průvodce; GRADA publishing 1998; Praha

[5] Hlavenka J. a kol. : Vytváříme WWW stránky a spravujeme moderní Website; Computer press 1997; Praha

[6] Internetové adresy:

URL: http://www.javascript.sk
URL: http://www.interval.cz
URL: http://prg.rkk.cz/dhtml
URL: http://www.kosek.cz
URL: http://svet.namodro.cz/w-dynamic