ČESKÁ ZEMĚDĚLSKÁ UNIVERZITA V PRAZE

PROVOZNĚ EKONOMICKÁ FAKULTA

INTERPRETACE KASKÁDNÍCH STYLŮ RŮZNÝMI WWW PROHLÍŽEČI

BAKALÁŘSKÁ PRÁCE

AUTOR: FILIP SKALKA

VEDOUCÍ: ING. PAVEL ŠIMEK

Prohlášení

Prohlašuji, že jsem bakalářskou práci na téma „Interpretace kaskádních stylů různými WWW prohlížeči“ vypracoval samostatně, pouze s použitím uvedených zdrojů literatury.

V Praze, 29.6.2007 .....

Filip Skalka

Poděkování

Rád bych poděkoval vedoucímu mé bakalářské práce Ing. Pavlovi Šimkovi za odborné připomínky a spolupráci během tvorby.

Interpretace kaskádních stylů různými WWW prohlížeči

Souhrn

Internetovým prohlížečům v současné době vévodí Microsoft Internet Explorer. Jeho dominantní postavení je zapříčiněno především tím, že je dodáván jako součást operačního systému Windows.

Avšak Internet Explorer čím dál tím více ztrácí tržní podíl na úkor dvou jeho největších konkurentů, Mozilly a Opery. Každý z prohlížečů má své výhody a nevýhody, horlivé zastánce i odpůrce. Mnoho kritiky si prohlížeče vysloužily také za rozdílné zobrazování stejných internetových stránek, spojených zejména s problematikou kaskádových stylů. S rozšiřujícím se používáním a možnostmi CSS vznikaly nové konflikty a stejně tak jejich řešení.

Klíčová slova

Internet Explorer, Mozilla, Opera, internetový prohlížeč, kaskádové styly

Interpretation of Cascading Style Sheets in different internet browsers

Summary

Microsoft Internet Explorer is a leader in the field of Internet browsers. Providing the browser as a part of Windows operating systems has mainly caused the dominant market position of Microsoft.

Nevertheless Internet Explorer is loosing more and more its market share at the expense of two most important competitors – Mozilla and Opera. Any of the browsers has got its advantages and disadvantages, as well as its keen advocates and opponents. The browsers have earned a lot of criticism because of different rendering of the same web pages. It largely applies web pages using Cascading Style Sheets.

With increased use and possibilities of CSS new confrontations have appeared together as well as their solutions.

Key words

Internet Explorer, Mozilla, Opera, internet browser, Cascading Style Sheets

1 . Obsah

2. Úvod

Původně byly internetové stránky navrženy s cílem spíše strukturovat informace na stránce, než jim dávat nějakou konkrétní grafickou prezentaci. Na prvních webových stránkách bohatě postačovalo, aby se dal odlišit nadpis od zbytku textu, případně se použilo zvýraznění část textu pomocí ztučnění. O to, jak přesně bude výsledná stránka zformátována, se staral internetový prohlížeč. Ten sám zvolil podle nastavení uživatele písmo, ale dnes běžné prvky, jako například přesné zalomení textu, bylo mimo jeho schopnosti. S rozvojem internetu začal být tento přístup naprosto nedostatečný. Požadavky firem na jejich webové prezentace spolu s potřebou snadné orientace na straně zákazníka, dávají vzniknout mnohým doplňkům a rozšíření jazyka HTML[1] nad rámec standardu. Dochází tak k roztříštění HTML, kdy se do něj implementuje velké množství funkcí a značek, které však fungují pouze v jednotlivých verzích konkrétních prohlížečů. Nastává dlouhodobě neudržitelný stav, kdy nové funkce do HTML vymýšlejí pouze výrobci prohlížečů a nikoliv autoři jazyka. Následuje degradace struktury dokumentů, poté, co se začne využívat tabulek pro pozicování prvků, se struktura ztrácí téměř úplně. To vede k tomu, že v dokumentu se vedle sebe nachází prvky, které spolu vůbec nesouvisejí, nebo je souvislý text naopak rozdělen do několika od sebe vzdálených míst. Stránky se tak správně zobrazují jen v prohlížeči, pro nějž jsou formátovány. Formátovací značky tvoří převážnou většinu datového objemu stránek.

Tato situace vede k vytvoření myšlenky oddělení obsahu od prezentace. Jazykem pro popis prezentace je CSS (Cascading Style Sheets – Tabulky kaskádových stylů). Kaskádové styly umožňují formátovat stránku, popsat styl jejích prvků a výslednou podobu, aniž by byl ovlivněn samotný obsah dokumentu. Lze díky nim využívat preciznější typografické možnosti,

[1] HyperText Markup Language

než umožňuje běžný jazyk HTML, a tak se přiblížit alespoň částečně k typografickým možnostem běžných DTP[2] aplikací, umožňují vytvářet stránky s podstatně úspornějším kódem a s konzistentnějším vzhledem, což je důležité zvláště u rozsáhlejších webových projektů. Je ovšem třeba mít na paměti, že ne všechny kaskádové styly jsou kompatibilní se všemi prohlížeči. Některé prohlížeče interpretují určité definice chybně a v důsledku toho může stránka vypadat nevzhledně až nečitelně.

[2] Desktop publishing

3 . Cíl práce a metodika

3.1. Cíl práce

Cílem této práce je nabídnout ucelený pohled na problematiku rozdílné interpretace kaskádových stylů nejpoužívanějšími internetovými prohlížeči, vysvětlit důvody nekompatibility a názorně ukázat možná řešení chyb.

3.2. Metodika práce

Využitím poznatků získaných studiem odborné literatury, konzultacemi se specialisty v oboru a zkušenostmi nabytými vlastní praxí autor prezentuje nejrozšířenější chyby v interpretaci kaskádových stylů a možná východiska pro tvorbu webových stránek tak, aby se správně zobrazovaly v různých prohlížečích.

První část práce se zabývá popisem kaskádových stylů, souvislostí CSS s internetovými stránkami a jejich vztahem k HTML kódu.

Hlavním cílem je co nejpřesněji popsat nestejné chování různých internetových prohlížečů při práci s kaskádovými styly a následně pro tyto případy prezentovat postupy řešení.

Jako prostředek k obcházení implementačních chyb existuje možnost pro některé prohlížeče použít jiné, přímo pro konkrétní prohlížeč určené deklarace nebo celé stylopisy.

Nejkrajnějším možným řešením je využití některého z mnoha CSS triků (CSS hacks). CSS triky obcházejí jednu implementační chybu prohlížeče využitím jiné chyby nebo nedokonalostí implementace.

Na závěr bude zhodnocena kvalita jednotlivých prohlížečů z hlediska správné interpretace kaskádových stylu, následně z toho vyvozeno doporučení vhodného prohlížeče webových stránek a shrnut obsah práce.

4 . Popis CSS

4.1. Co je to CSS

Kaskádové styly jsou standardem konsorcia W3C, který umožňuje ovlivňovat typografický vzhled našeho dokumentu uloženého v HTML. S jejich pomocí dosáhneme toho, že se HTML používá k tomu, k čemu bylo určeno, tedy k popisu struktury textu. V podstatě jsou kaskádové styly pravidla, složená ze dvou částí – za prvé, kde se mají použít, a za druhé, co mají na výsledné stránce ovlivnit. První část nazýváme selektorem a druhou část definicí, která se skládá z jedné či více středníkem oddělených deklarací.

Existují různé způsoby, jak propojit styl s dokumentem:

- **Řádkový styl** – přímo v otvíracím tagu formátovaného elementu pomocí atributu style="...".
- **Interní stylový dokument** – pomocí "stylopisu" v hlavičce stránky. Stylopis je jakýsi seznam stylů, ve kterém je obecně napsáno, co má být jak zformátováno. Do stránky se stylopis píše mezi tagy <style> a </style>.
- **Externí stylový dokument** – jedná se o soubor *.css, na který se stránka odkazuje tagem <link>. V souboru je umístěný stylopis. Hlavní výhoda spočívá v možnosti odkázat na jeden takový soubor libovolný počet stránek, které pak používají stejný styl.
- **Externí styl volaný z dokumentu CSS** – umožňuje importovat stylopis z externího souboru CSS, k čemuž slouží příkaz @import. Ten je možné zapsat nejen do hlavičky stránky, ale také na začátek interního či externího stylového dokumentu.

V současné době se převážně používají následující verze CSS:

- **CSS 1 –** umožňuje nastavení vlastností písma, okrajů, barev, okrajů, pozadí atp.

- **CSS 2.1** – obsahuje vše z CSS1 a přidává elementy pro absolutní pozicování, automatické číslování, zalomení stránky, text psaný zprava doleva a další.
- **CSS 3** – tato verze se nachází ve vývoji, předpokládá se, že s CSS 3 se dostane formátování stránek na úroveň aplikací DTP. Bude kromě schopností předchozí verze obsahovat například sloupcovou sazbu, oboustranné obtékání obrázků, správu barev, nové textové efekty atp.

4.2. Standardy

Webové standardy definuje od roku 1994 organizace s názvem World Wide Web Consortium (W3C) [1]. Oficiálním cílem konsorcia je vedení technické evoluce webu, k čemuž schválilo během svého působení přes 90 standardů a doporučení, včetně specifikace kaskádových stylů.

Podle platných standardů W3C by měly být pro vizuální formátování používány výhradně kaskádové styly CSS1 a CSS2. Pomocí nich zajistíme oddělení obsahové struktury dokumentu HTML od formátovacích prvků, které mohou být uloženy v samostatném souboru CSS, ten pak může být společný pro všechny stránky projektu. Bude-li takto zformátovaný dokument načten prohlížečem, podporujícím kaskádové styly, bude renderován jako plně grafický. Naopak, při načtení v prohlížeči s nedostatečnou či vůbec žádnou podporou kaskádových stylů, uvidíme pouze neformátovaný, přesto však stále strukturovaný dokument. V obou prohlížečích dosáhneme kýženého výsledku, čímž je přehledně uspořádaný a čitelný dokument s intuitivní navigací.

Co je ovšem podstatné při dodržení standardů, je kompatibilita. A kompatibilita zpětná, pro zobrazení neformátované strukturované verze ve starších prohlížečích, i dopředná, pro zobrazení plně graficky formátované verze v moderních současných i budoucích prohlížečích.

Postupně se do projektu W3C začínají zapojovat i jednotliví výrobci prohlížečů a zavádět do praxe dohodnuté standardy. Implementace funkcí CSS

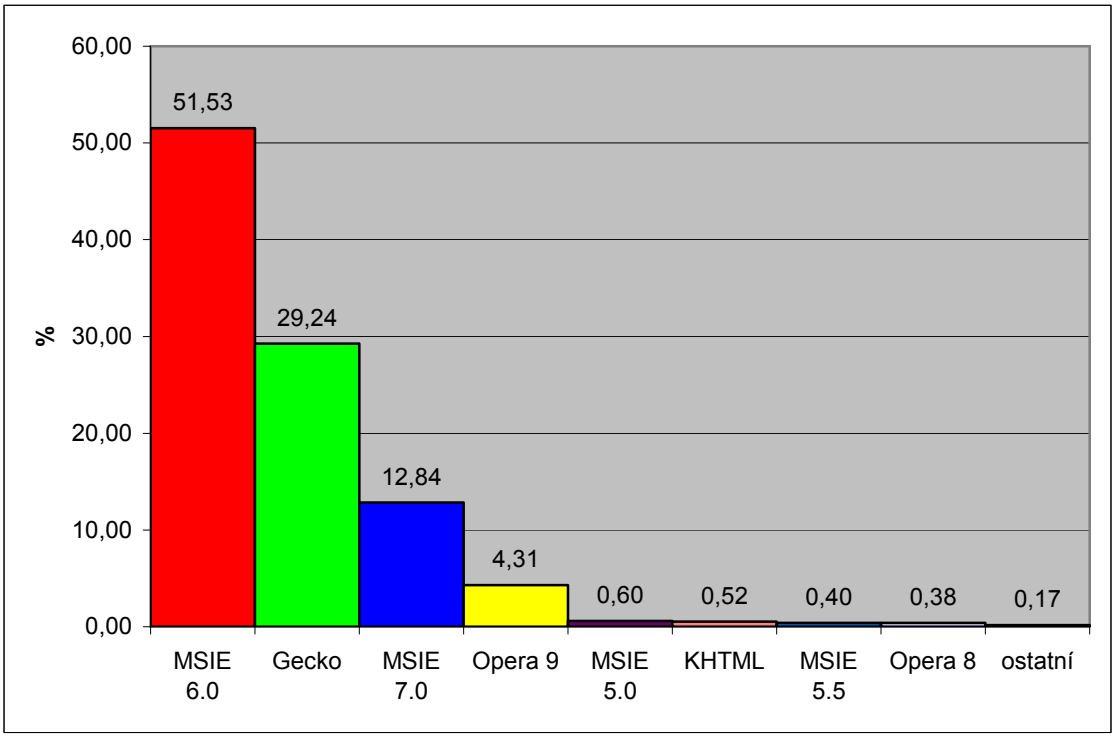
však probíhá u každého prohlížeče odlišným tempem a někdy i odlišným způsobem.

5 . Rozdíly v interpretaci CSS webovými prohlížeči

Nekompatibilita je uváděna jako jediná nevýhoda CSS [2]. Na následujících stránkách najdete přehled vlastností CSS, u nichž se liší zobrazení nejběžnějšími prohlížeči. V šesté kapitole budou uvedeny některé postupy řešení v případě, že chceme určité funkce CSS použít pro nekompatibilní prohlížeče.

5.1. Relevantní prohlížeče

V dalším textu se zabýváme těmito prohlížeči: Microsoft Internet Explorer pro Windows verze 4.0, 5.0, 5.5, 6 a 7, Mozilla (a ostatní prohlížeče založené na jádře Gecko, pro jednoduchost souhrnně nazývanými Mozilla, v praxi se jedná zejména o Firefox 1 a 2) a Opera – verze 4.02, 5, 6, 7, 8 a 9. Následující graf zobrazuje procentuální zastoupení prohlížečů na trhu:



Graf 1: Zastoupení prohlížečů na internetu

Zdroj: [3]

Režimy prohlížeče

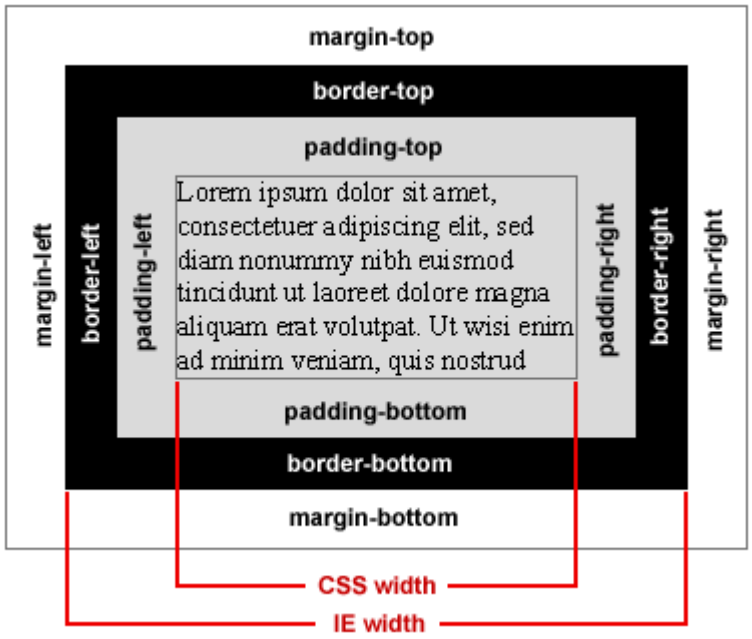
Při řešení problematiky kompatibility prohlížečů je důležité zmínit, že se jejich výrobci rozhodli zabudovat do novějších verzí svých programů (IE 6+, Opera 7+, Mozilla) takzvané režimy zpracování:

- **standardní režim** – prohlížeč pracuje co nejpřesněji podle specifikace jazyků (X)HTML, CSS atd. V tomto režimu se méně tolerují chyby, model vykreslování se chová podle specifikace norem W3C.
- **manýristický (quirk) režim** – prohlížeč simuluje chybné chování starých verzí MSIE nebo Netscape, se všemi jejich chybami. Napodobuje jejich chybné postupy a nepřípustné odchylky od specifikací.

Z důvodů zpětné kompatibility jsou tyto prohlížeče schopny zpracovat oba modely. Podle potřeby použijí jeden z nich, podle použitého značkovacího jazyka. Pokud je stránka vytvořená v jazyce HTML 3.2, HTML 4 (Transitional), nebo není-li použitý jazyk určen deklarací DOCTYPE, použije se quirk režim vykreslování. V případě, že je stránka vytvořena striktní verzí HTML 4 (Strict), nebo jazykem XHTML, použije se standardní režim zobrazování.

5.2. Výpočetní model rozměrů rámů

Vůbec nejznámější implementační chybou je chybný box model Internet Exploreru. Specifikace CSS považuje vlastnost prvku width za šířku jeho obsahu. Celková viditelná šířka tak vznikne součtem vlastností width, padding (výplň) a border (rámeček). Internet Explorer však vlastnost width interpretuje jako celkovou viditelnou šířku. Skutečná velikost obsahu je pak menší o velikosti rámečku a výplně.



Obrázek 1: Rozměry boxu podle CSS specifikace a podle IE

Zdroj: [4]

Z obrázku je vidět, jak je vlastnost width specifikována a jak je chápána Internet Explorerem, verzemi 4, 5, 5.5, 6 a 7. U šesté a sedmé verze IE chyba vzniká jen v režimu zpětné kompatibility (quirk).

5.3. Standardní vlastnosti, které IE neumí

Prohlížeče Mozilla a Opera se vynikají velmi dobrou podporou vlastností CSS. Naopak Internet Exploreru se v této oblasti dostává nemalé kritiky. Následující vlastnosti patří k nejdůležitějším, které Internet Explorer neumí.

5.3.1. Obecné funkce CSS

- Import – MSIE 4 podporuje pouze syntaxi import url(…), nikoliv však import "…"
- Selektory tříd – IE 5.x chybně interpretuje vícenásobné třídy

- Selektory potomek (A>B) a sousední sourozenec (A+B) – IE 4-5 ignoruje > a + a chybně interpretuje oba zápisy jako A B
- Escape-sekvence – IE 4-5 nepodporují správně escape-sekvence v řetězcích
- :hover – do verze 7 lze použít pouze u odkazů
- :before, :after – pseudotřídy sloužící k automatickému zařazování prvků webové stránky řízené pomocí CSS – IE neumí

5.3.2. Vlastnosti rámů

- Výplň a rámečky u řádkových prvků – není v IE 4-5 podporováno
- Border-style – IE 4-5 nepodporuje tečkovaný ani čárkovaný okraj, použijí plnou čáru. IE 6 podporuje pouze čárkovaný okraj, který použije i u tečkovaného
- Width a height – kromě IE 6 a 7 ve standardním režimu používá IE odlišný výpočetní model (detaily v kapitole 5.1)
- Min-width, max-width – nastavení minimální nebo maximální šířky prvku webové stránky není možné v IE kromě verze 7 ve standardním módu
- Min-height, max-height – nastavení minimální nebo maximální výšky prvku. V IE 6 funguje pouze min-height, a to jen u tabulkových prvků uvnitř tabulky s parametrem table-layout: fixed (možné řešení se nachází v kapitole 6.2.10). Opraveno ve verzi 7
- Position:fixed – IE do verze 7 neumožňuje fixovat pozici jednoho prvku k rodičovskému prvku webové stránky
- Display: table – vytvoření uživatelské tabulky bez potřeby užití značek HTML pro tabulky nepodporuje žádná verze
- Float – starší verze IE (4-5) zpracovávají plovoucí prvky chybně, určité problémy jsou ale i ve verzích novějších

- Overflow: visible – místo aby obsah přetékal rám, je natažen aby se do něj obsah vešel. Opraveno ve verzi 7
- Opacity – nastavení úrovně průsvitnosti prvku, není podporováno do verze 5.5

5.3.3. Ostatní funkce

- Background-repeat – Internet Explorer verze 4 opakuje obrázek pouze směrem dolů a doprava
- Generování obsahu – vlastnosti content a quotes nejsou podporovány.
- Cursor: pointer – IE podporuje pointer až od verze 6

Zdroj: [5,6]

Kvůli ne zcela bezchybné implementaci navíc vykreslí MSIE v určitých případech zobrazovanou stránku chybně, přestože jsou použity jen ty vlastnosti CSS, které MSIE jinak podporuje. Popis těchto problémů a jejich řešení je uvedeno v kapitole 6.2.

5.4. Nestandardní vlastnosti navíc

5.4.1. Nestandardní vlastnosti Mozilly

Prohlížeč Mozilla a ostatní, založené na jádře Gecko, podporují řadu vlastních vlastností CSS. Mnohé z nich jsou inovativní, po kterých tvůrci webů již dlouho touží. Bohužel jsou použitelné jen pro uživatele těchto prohlížečů, což minimalizuje jejich praktické využití. Avšak mnohými funkcemi se inspirovali tvůrci CSS3, ve kterém jim podobné v budoucnu najdeme.

Tyto vlastnosti, začínající na -moz-, umožňují například přizpůsobení vzhledu stránky vzhledu nastaveném v operačním systému, kulaté okraje boxu, jeho orientaci apod.

Více na [7]

5.4.2. Stylování scrollbarů (IE a Opera)

Internet Explorer od verze 5.5 a Opera od verze 7.2 nabízí možnost definovat barvy rolovacích lišt. Nastavit vlastní barvu lze každému prvku scrollbaru, či použít základní obarvení, kdy se celý posuvník stylizuje do jedné barvy použitím následujícího zápisu:

```
<style>
body, html  { scrollbar-base-color: red; }
</style>
```

Příklad 1: Stylování scrollbarů

6 . Postupy řešení nekompatibility CSS s některými prohlížeči

6.1. Různé CSS pro různé prohlížeče

V případě problémů s nekompatibilním zpracováním kaskádových stylů můžeme dosáhnout kýženého zobrazení stránek ve všech prohlížečích tak, že vytvoříme dva, případně několik CSS souborů, pro různé prohlížeče a jejich verze. Obvykle se setkáme s použitím jednoho základního pro všechny prohlížeče a druhým doplňkovým, výhradně pro MSIE.

6.1.1. Podmíněné komentáře

Prvním jednoduchým řešením je použití takzvaných podmíněných komentářů. Jedná se o novou sadu značek, použitelnou v IE 5+. Pomocí nich můžeme rozpoznat, zda se jedná o prohlížeč IE, dokonce i kterou konkrétní verzi IE používáme. Ostatní prohlížeče tuto formulaci chápou jako komentář, tudíž jí ignorují.

```
<html>
<head>
<link rel="stylesheet" type="text/css" media="all"
href="style.css"/>
<!--[if IE ]>
<link rel="stylesheet" type="text/css" media="all"
href="./styleIE.css" />
<![endif]-->
</head>
<body>
</body>
</html>
```

Příklad 2: Podmíněné komentáře

Zdroj: [8]

6.1.2. @import

Pro odříznutí IE4 od stylu lze využít jeho neznalost syntaxe import "…". Tato stará verze podporuje pouze formulaci import url(…).

```
<style type="text/css">
@import url("styleIE.css"); // Tento soubor načte i IE4
@import "style.css"; //style.css IE4 načíst nedokáže, ostatní ale
ano
</style>
```

Příklad 3: Odříznutí IE 4 pomocí @import

Podobně lze využít metodu @media, díky které se nám podaří oříznout i IE5.0. Zdroj: [9]

6.1.3. Podtržítkový hack

Velice elegantní řešení pro rozdílné definování Internet Exploreru a ostatním prohlížečům vychází ze specifikace CSS, podle které mohou identifikátory obsahovat i znak "_" (podtržítko). Prohlížeče by zároveň měli ignorovat definování jim neznámých vlastností. Jenže prohlížeče MSIE 5 a 6 podtržítko na začátku identifikátorů ignorují a chovají se jako by tam nebylo, tudíž zpracují i definici, kterou ostatní prohlížeče přeskočí jako neznámou vlastnost.

```
</style>
#sample {
   position: fixed;
   _position: absolute;
   }
</style>
```

Příklad 4: Podtržítkový hack

V tomto příkladu bude pro MSIE do verze 6 definováno absolutní pozicování namísto pro něj neznámého fixovaného. To bude použito u všech ostatních prohlížečů.

Internet Explorer 7 již chápe podtržítko i pomlčku za součást jména vlastnosti, ignoruje ovšem stále mnoho jiných znaků, například !@#$%^&*<>()[]/=.

Zdroj: [10]

6.1.4. Zpětné lomítko

Vložením zpětného lomítka do názvu vlastnosti spolehlivě odřízneme IE 5.5 a starší od ostatních prohlížečů.

```
</style>
#sample {
   vlastnost: hodnota; /*pro všechny prohlížeče*/
   v\lastnost: hodnota; /*pro všechny kromě IE 5.5 a nižších*/
  }
</style>
```

Příklad 5: Použití zpětného lomítka

6.1.5. Tantek Çelik

Tantek Çelik se nazývá hack, pomocí kterého lze přerušit zpracování definice v IE 5.5 a starších, zatímco ostatní prohlížeče pokračují. Hack funguje tak, že nastavíme vlastnost voice-family na hodnotu \"}\" a následně ji vrátíme na původní hodnotu. Kvůli chybě parseru v IE tento prohlížeč interpretuje první řádek hacku jako ukončení celé definice a její zpracování zde přeruší.

Příklad použití Tantek Çelik hacku pro řešení Box model bugu (viz kapitola 5.1).

```
<style type="text/css">
#prvek {
margin : 10px;
border : 10px;
padding : 40px;
width : 400px;
voice-family : "\"}\""; /* zde přeruší IE5.5 a starší definici
identifikátoru */
voice-family : inherit; /* nastavení vlastnosti na výchozí hodnotu
*/
width : 300px; /* nastavení správné šířky pro ostatní prohlížeče */
}
</style>
```

Příklad 6: Tantek Çelik hack

Zdroj: [11]

6.1.6. Diakritický CSS trik

Diakritický CSS trik spočívá v použití diakritiky (kupodivu s „š“ a „ž“ problémy nejsou) v komentářích před CSS definicí. Internet Explorer, kromě verze 7, vše za tímto řádkem ignoruje až do dalšího komentáře bez diakritiky nebo prázdného komentáře.

```
<style>
p {color:red} // Tuto definici MSIE použije

/* MSIE čistič */
p {color:green} // Tyto dvě definice MSIE nepoužije

/**/
p {background:black}  // Tuto definici MSIE opět použije
</style>
```

Příklad 7: Diakritický CSS trik

Výsledkem tohoto příkladu bude v Internet Exploreru červené písmo na černém pozadí, v prohlížečích Opera a Mozilla bude písmo zelené (platí, že je použita pozdější definice).
Zdroj: [12]

6.1.7. Holly Hack

Holly hack se obvykle skládá ze dvou částí. V první se využije vlastnost některých parserů, že se znak za zpětným lomítkem se neinterpretuje znak se speciálním významem. Pokud se ale nachází zpětné lomítko v komentáři, tuto vlastnost ztrácí. Ne však u IE určeným pro Macintosh, čímž před ním schováme zbytek kódu až do dalšího ukončení komentáře. Druhá část hacku použije formulaci * html #název, kterou naopak dokáží přečíst pouze prohlížeče IE. Vlastní trik holly hacku spočívá v nastavení výšky prvku na malou hodnotu, IE si ho pak sám roztáhne, pracuje s vlastností height tak, jako by to byla vlastnost min-height. Tento hack se často využívá při řešení problémů s plovoucími prvky.

```
<style>
/* Schovat před IE-Mac \*/
* html #sample
   {
   height: 1%;
   }
/* Obnovit IE-Mac */
</style>
```

Příklad 8: Holly Hack

Zdroj: [13]

6.1.8. !important

Moderní prohlížeče znají funkci !important, pomocí které se nastaví vyšší priorita definice v CSS, kterou následně definováním stejného prvku nepřebijeme. IE v quirk režimu !important neznají.

```
<style>
#sample {
width: 100px !important;  /* Tento řádek nastaví konečnou šířku pro
Mozillu a Operu */
width: 200px; /* IE !important neznají a tento řádek přebije
předchozí */
}
</style>
```

Příklad 9: Použití !important

6.2. Řešení chyb a neimplementovaných vlastností v IE

6.2.1. Matrjoška trik

Matrjoška trik, pojmenovaný podle ruské hračky – dřevěných panenek, které se do sebe dají skládat, spočívá v nahrazení jednoho elementu několika do sebe vnořenými elementy, zpravidla neutrálními blokovými elementy div. Využívá se toho, že prvek div se bez CSS nijak neformátuje a jeho přítomnost v nestylovaném dokumentu není patrná. Tento trik nám umožňuje řešit některé problémy nekompatibility Internet Exploreru, například Double margin bug (viz kapitola 6.2.3) nebo Box model bug.

Jako příklad si ukážeme využití matrjošky právě při řešení Box model bugu:

Původní prvek umístíme ještě do jednoho DIVu. Tomuto vnějšímu prvku nastavíme pouze width. Padding a border nastavíme jako nulové. Vnitřnímu prvku nastavíme potřebný padding a border, width necháme na výchozí automatické hodnotě, případně ji nastavíme na width:auto.

Dosáhneme tak toho, že u vnějšího prvku zadaná šířka pouze ohraničuje dostupný prostor vnitřního prvku. Ten má nastavený potřebný padding a border, díky automatické šířce se pak roztáhne do šířky ohraničené prvkem vnějším.

```
<style>
#obalka {width: 350px; padding: 0 }
#obsah { padding: 8px; border: 2px solid black; background: white}
</style>
...
<div id="obalka">
    <div id="obsah">
    text
    </div>
</div>
```

Příklad 10: Matrjoška trik

Zdroj: [14]

6.2.2. Peekaboo bug

Peekaboo bug se projevuje tak, že při vykreslení Internet Explorerem 6 se nezobrazí část textu na stránce. Text na stránce však je a lze myší označit, případně se zobrazí při zúžení okna prohlížeče. Za určitých okolností se totiž pozadí nějakého nadřazeného bloku vykreslí později, než blok v něm vnořený. Tím pádem pak není text v onom vnořeném bloku vidět.

Tato chyba nastane, pokud dokument obsahuje box, který nemá určenou šířku a má definovánu libovolnou barvu pozadí. Tento box obsahuje jiný box, který však je plovoucí, tzn. obsahuje parametr float. Plovoucí box je obtékán odstavcovým textem, a zároveň je dolní hrana okraje tohoto plovoucího boxu níže, než dolní hrana okraje odstavcového textu. Tento text je následován libovolným elementem s definovanou vlastností clear. Mezi plovoucím boxem a boxem ve kterém se nachází se nesmí nacházet jiný prvek s definovanou výškou nebo šířkou.

```
<style type="text/css">
#obal { padding : 1px; background-color : gray; }
#floated { width : 50px; height : 50px; float : left; background-
color : #cccccc; }
#cleared-text { clear : left; }
</style>
```

```
<div id="obal">
<div id="floated">
</div>
mizející text
<div id="cleared-text"></div>
</div>
```

Příklad 11: Peekaboo bug

Obrázek 2: Peekaboo bug

Řešení je v tomto případě mnoho:

- Nadřazenému boxu nastavíme šířku nebo výšku, např. width: 100%.
- U obtékajícího odstavcového textu nastavme vlastnost position na hodnotu relative.
- Můžeme-li, zrušíme nadřazenému boxu barvu pozadí.
- Aplikace Holly Hacku:

```
<style type="text/css">
#obal { padding : 1px; background-color : gray; }
/* \*/  * html #obal { height: 1px; }  /* */
#floated { width : 50px; height : 50px; float : left; background-
color : #cccccc; }
#cleared-text { clear : left; }
</style>
```

Příklad 12: Řešení Peekaboo bugu

Zdroj: [15]

6.2.3. Double margin bug

Double margin bug se projevuje v IE u plovoucích prvků tak, že vlevo plovoucí prvek má dvojnásobně velký levý vnější okraj (margin), vpravo plovoucí prvek má dvojnásobný pravý okraj. Tato chyba byla opravena ve verzi IE 7.

```
<style>
#float-box { float: left; margin-left: 20px; width: 100px; height:
100px; background-color: #000000 }
</style>
```

Příklad 13: Double margin bug

Tento plovoucí box se v Internet Exploreru zobrazí s levým okrajem (margin-left) 40px.

Double margin bug se obvykle řeší přidáním deklarace display: inline, použít se dá i matrjoška hack (kapitola 6.2.1).

```
<style>
#float-box { float: left; margin-left: 20px; width: 100px; height:
100px; background-color: #000000; display: inline; }
</style>
```

Příklad 14: Řešení Double margin bugu

Zdroj: [16]

6.2.4. Float bug

Float bug způsobí nesprávné zobrazení plovoucího boxu ve všech verzích Internet Exploreru. Pokud máme dva boxy, kdy jeden z nich je plovoucí a v logické struktuře dokumentu je umístěn před druhým boxem, zobrazí se vedle druhého boxu, místo toho aby ním byl obtékán. Chyba se vyskytuje i v IE 7.

```
<style>
#float { background-color: #ff0000; float: left; width: 50px;
height: 50px; }
#box { width: 200px; }
</style>
…
<div id="float"></div>
<div id="box">Obsah</div>
```

Příklad 15: Float bug

Jednoduchým řešením je vložit oba elementy do nového obalového a tomu přiřadit požadovanou šířku původního neplovoucího boxu.

```
<style>
#obal { width: 200px; }
#float { background-color: #ff0000; float: left; width: 50px;
height: 50px; }
</style>
…
<div id="obal">
<div id="float"></div>
<div id="box>Obsah</div>
</div>
```

Příklad 16: Řešení float bugu

6.2.5. 3px bug

Dalším problémem s plovoucími prvky je 3px bug. Pokud máme v obalovém boxu další plovoucí box a text, bude text v IE do verze 6 včetně o 3 pixely odsazen.

```
<style>
#wrap { width: 200px; }
#float { background-color: #ff0000; float: left; width: 50px;
height: 50px; }
#box { background-color: #0000ff; }
</style>
```

```
<div id="wrap">
<div id="float"></div>
<div id="box">obsah</div>
</div>
```

Příklad 17: 3px bug

Možné řešení je nastavit pouze Internet Exploreru 6 a starším zápornou hodnotu levého okraje margin-left: -3px;

6.2.6. Nezobrazení vertikálního posuvníku

Tato chyba se projevuje v IE od páté do šesté verze nezobrazením vertikálního posuvníku, při přečtení obsahu absolutně pozicovaného prvku, který zasahuje mimo viditelnou část stránky.

Podmínky pro vznik bugu jsou následující: existují dva do sebe vnořené boxy, vnější, prvek A, nemá definovány rozměry a má nastaveno relativní pozicování. Vnořený box B je pozicovaný absolutně. Nachází-li se v prvku B obsah, který se nevejde do okna prohlížeče, IE obrazí vertikální posuvník a zbylou část obsahu se nelze dostat.

```
<style>
#A { position: relative; }
#B { position: absolute; }
</style>
…
<div id="A">
<div id="B"> obsah prvku B </div>
</div>
```

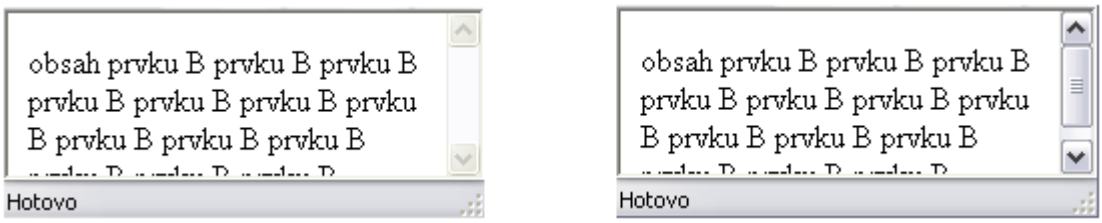
Příklad 18: Nezobrazení vertikálního posuvníku

Tato chyba se dá napravit nastavením šířky nebo výšky vnějšímu prvku. Můžeme zde využít tzv. Holly Hacku, kdy nastavíme width či height na hodnotu 1px

(můžeme použít i 1%) a prohlížeč si box následně roztáhne na potřebnou velikost.

```
<style>
#A { position: relative; }
/* \*/ * html #A { height: 1px; } /* */
#box { position: absolute; }
</style>
```

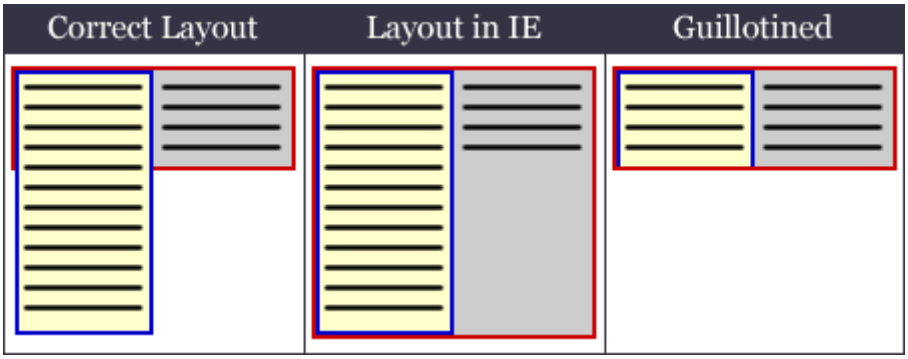
Příklad 19: Řešení nezobrazení vertikálního posuvníku



Obrázek 3: Nezobrazení vertikálního posuvníku

6.2.7. *Guillotine bug*

Guillotine bug je chyba v Internet Exploreru 6, která uřízne spodní část plovoucího elementu po přejetí myší přes odkazy. Tyto odkazy musí mít nadefinovanou změnu vlastností při najetí na ně kursorem, např. změna pozadí, výplně, stylu textu, okrajů…, a zároveň se nacházet v obalovém boxu, po plovoucím elementu. Poté, co najedeme na třetí odkaz, plovoucí box ořízne na výšku neplovoucího obsahu.



Obrázek 4: Guillotine Bug

Zdroj: [16]

```
<style>
#box { float: left; border: 1px solid #000000; width: 100px; }
a:hover { padding: 5px; }
</style>
…
<div id="obal">
<div id="box">Plovoucí text, který je delší než neplovoucí
obsah</div>
<a href="#">Odkaz 1</a><br />
<a href="#">Odkaz 2</a><br />
<a href="#">Odkaz 3</a><br />
<a href="#">Odkaz 4</a><br />
<a href="#">Odkaz 5</a>
</div>
```

Příklad 20: Guillotine bug

Řešením je umístění prázdného elementu za nebo na konec obalového boxu.

Jiné řešení se nabízí v případě, že obalový box může být roztažen s výšku boxu plovoucího. V tom případě umístíme obsah obalového boxu do dalšího elementu, na který aplikujeme Holly hack.

6.2.8. Italics bug

Italics bug způsobuje v Internet Exploreru ve všech verzích, že box, ve kterém je text psaný kurzívou, se neočekávaně rozšíří, přestože by měl volně přetékat přes okraj. Zároveň jsou písmena zleva ořízuta.

```
<style>
#box { width: 200px; font-size: 200%; font-style: italic; text-
align: justify; }
</style>
```

Příklad 21: Rozšíření boxu při Italics bugu

Takto definovaný box bude mít v Internet Exploreru šířku 206px. Pokud budeme chtít použít text psaný kurzívou na celou stránku, dojde tak ke zbytečnému zobrazení horizontálního posuvníku.

```
<style>
#box { width: 100%; font-style: italic; }
</style>
```

Příklad 21: Zobrazení horizontálního posuvníku při Italics bugu

Abychom se těmto problémům vyhnuli, je třeba nastavit pro IE 4 a 5 vlastnost overflow na hodnotu hidden, ostatním verzím Internet Exploreru na hodnotu visible. Poté pomocí Holly hacku nastavíme výšku boxu na 1px či %, protože ke správné funkčnosti je potřeba definovat alespoň jeden rozměr. Tím se zamezí přetékání znaků z boxu. Komentářovým hackem nejprve odřízneme IE pro Macintosh, potom kombinací selektorů * html všechny jiné prohlížeče než IE, nakonec pomocí lomítka v názvu vlastnosti rozlišíme definice pro Internet Explorer 4 a 5, který takovou vlastnost nezpracuje, a pro vyšší verze IE.

```
<style>
#box { width: 100%; font-style: italic; }
/* \*/
* html #box { overflow: hidden; o\verflow: visible; he\ight: 1%; }
/* */
</style>
```

Příklad 22: Řešení Italics bugu

Zdroj: [17]

6.2.9. Missing first letter bug

Máme dva do sebe vnořené blokové elementy. Ve vnitřním je nějaký text a má nastaveny vlastnosti position na hodnotu relative a letter-spacing na libovolnou

hodnotu. Když si takto definovanou stránku prohlédneme v Internet Exploreru 5.5 zjistíme, že se nevykreslilo první písmeno, ačkoliv na něj je místo.

```
<style>
#box { position: relative; letter-spacing: 10px; }
</style>
…
<div id="obal">
<div id="box">Missing first letter bug</div>
</div>
```

Příklad 23: Missing first letter bug

Řešení je více, např. můžeme nastavit vnějšímu elementu horní okraj (top border), horní výplň (top padding), přidat jednomu z boxů definici šířky či výšky (Holly Hack), nebo odstranit definici relativního pozicování.

Řešení pomocí Holly Hacku:

```
<style>
/* \*/ * html #obal { height: 1px; } /* */
#box { position: relative; letter-spacing: 10px; }
</style>
```

Příklad 24: Řešení Missing first letter bugu

6.2.10. Hack pro "min-height"

O tom, ze Internet Explorer neumí, až na poslední verzi 7, vlastnost min-height již byla zmínka dříve. Chceme-li však minimální výšku elementu přesto použít, dosáhneme toho pomocí jednoduchého hacku.

Nejprve nastavíme min-height a height na požadovanou minimální výšku. Pokud by byl obsah delší, IE box prodlouží. Následně využijeme neznalosti selektorů s atributy Internet Explorerem a pro ostatní prohlížeče nastavíme height zpět na automatickou hodnotu.

```
<style>
.box { min-height:100px; height:100px; }
div[class] .box { height:auto; }
</style>
```

Příklad 25: Hack pro "min-height"

6.2.11. :hover

Pseudotřídu :hover je možné v Internet Exploreru do verze 7 použít pouze u odkazů. Pro dosažení stejného efektu dosáhnout i u jiných prvků je potřeba použít náhradu v podobě JavaScriptu. Tato metoda má tu výhodu, že HTML kód zůstane nezměněn.

Pro vybraný element se nastaví základní styl (pokud se nad elementem nenachází myš) jako kdykoli předtím. Styl použitý při zvýraznění, je-li kurzor myši nad elementem, se pro prohlížeče, které to umí, definuje prostřednictvím pseudotřídy :hover. Kvůli IE se stejná definice použije i pro třídu, pro přehlednost pojmenovanou taktéž hover. Žádné prvky v HTML dokumentu nemají třídu hover nastavenou v atributu class, dynamickou změnu třídy podle polohy myši v IE zajišťuje právě JavaScript. Připojení JavaScriptového kódu se provede v zápisu CSS pomocí vlastnosti behavior.

Vytvoříme externí soubor s následujícím kódem:

```
<PUBLIC:ATTACH EVENT="onmouseover" ONEVENT="hoverRollOver()" />
<PUBLIC:ATTACH EVENT="onmouseout" ONEVENT="hoverRollOff()" />
<script language="JavaScript" type="text/javascript">
function hoverRollOver() {
 element.origClassName = element.className;
 var tempClassStr = element.className;
 tempClassStr += "Hover";
 tempClassStr = tempClassStr.replace(/s/g,"Hover ");
 tempClassStr += " hover";
 element.className = element.className + " " + tempClassStr;
}
```

```
function hoverRollOff() {
element.className = element.origClassName;
}
</script>
```

Příklad 27: Řešení neznalosti :hover Internet Explorerem pomocí JavaScriptu

Tento soubor pojmenujeme například „hover.htc“. Do CSS vložíme:

```
p {styl pro element p}
p.hover, p:hover {color: red;} /*třída pro IE, pseudotřída pro
ostatní prohlížeče*/
p {behavior: url(hover.htc);}
```

Příklad 28: Řešení neznalosti :hover Internet Explorerem ve stylopisu

Zdroj: [18]

6.2.12. Mizející obsah – Zmizík

V Internet Exploreru 7 ve standardním módu dojde ke zmizení plovoucího obsahu bloku, pokud je daný blok relativně pozicovaný, nemá určené rozměry a neplave. Plovoucí obsah následuje čistič v podobě vodorovné čáry (hr).

```
<style type="text/css">
    div { position: relative; }
    p   { float: left; }
    hr  { clear: both; }
</style>
Obsah před inkriminovaným boxem.
<div>
    <p>Tento text není v IE7 vidět</p>
    <hr>
</div>
Obsah za inkriminovaným boxem.
```

Příklad 29: Mizející obsah

Řešením je nastavení relativně pozicovanému bloku alespoň jeden rozměr, výsledek může vypadat například následovně:

```
<style type="text/css">
div { position: relative; min-height: 1px; }
    p   { float: left; }
    hr  { clear: both; }
</style>
```

Příklad 30: Řešení mizejícího obsahu

Zdroj: [19]

6.3. Chyby prohlížečů založených na jádře Gecko

V prohlížečích založených na jádře Gecko byly doposud objeveny pouze dva bugy. Oba již byly opraveny a v posledních verzích prohlížečů se nevyskytují.

6.3.1. Mozilla Clearing Bug

Tato chyba se vyskytovala při splnění určitých podmínek, pokud po sobě následovaly plovoucí blokový element zarovnaný doleva, plovoucí blokový element zarovnaný doprava a element s nastavenou vlastností clear:right (případně plovoucí blokový element zarovnaný doprava, plovoucí blokový element zarovnaný doleva a element s nastavenou vlastností clear:left). K chybnému zobrazení docházelo v případě, pokud se plovoucí elementy s pevně nastavenou šířkou nevešly vedle sebe. Má-li poslední element vlastnost clear:both namísto clear:right (resp. left) k chybě nedojde.

Žádné řešení s výjimkou vyhýbání se uvedené konstrukci není známé. Vzhledem k tomu, že tento bug byl opraven již ve verzi Firefox 1.0, není ani příliš potřeba.

6.3.2. Gecko Shifting Gaps 'n Overlaps

K chybě dochází, pokud se vertikálně uspořádané blokové elementy nachází uvnitř obalového divu, kterému je nastavena vlastnost line-height na nějakou procentuální hodnotu různou od 100%.

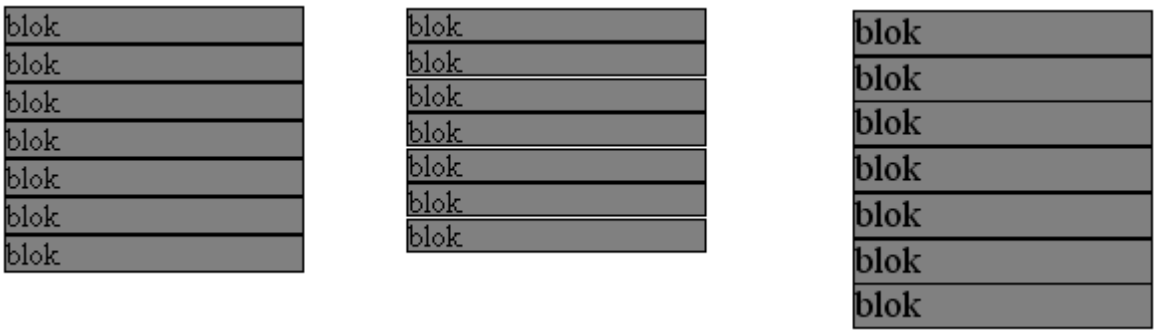
Chyba se projevuje občasnými (ale pravidelně rozmístěnými) mezerami mezi jednotlivými bloky nebo naopak překrýváním okrajů těchto bloků.

Řešením problému je nenastavovat obalovému divu vlastnost line-height na procentuální hodnotu různou od 100%. Taková hodnota nesmí být ani zděděná z nadřazeného elementu.

Pokud se uvedený příklad zobrazuje bezchybně v prohlížeči, o kterém je známo, že touto je touto chybou postižen (např. Mozilla Firefox 1.0), stačí změnit velikost písma a chyba se projeví.

```
<style>
#wrapper {width: 150px;
          line-height: 97%;  /* Cokoliv mimo 100% */}
.polozka {background-color: gray;
          border: 1px solid black; }
</style>
…
<div id="wrapper">
 <div class="polozka">blok</div>
 <div class="polozka">blok</div>
 <div class="polozka">blok</div>
 <div class="polozka">blok</div>
 <div class="polozka">blok</div>
 <div class="polozka">blok</div>
 <div class="polozka">blok</div>
</div>
```

Příklad 31: Gecko Shifting Gaps 'n Overlaps



Zdroj: [16]

Správné zobrazení Mezery mezi bloky Překrytí bloků

Obrázek 5: Gecko Shifting Gaps 'n Overlaps

7. Závěr

Cílem práce bylo nabídnout ucelený pohled na problematiku rozdílné interpretace kaskádových stylů nejpoužívanějšími internetovými prohlížeči, vysvětlení důvodů nekompatibility a názorné ukázání možných řešení chyb.

Výše popsané postupy vedou k odstranění většiny známých chyb v prohlížečích a umožňují vytvoření internetových stránek s využitím technologie CSS tak, aby se ve všech běžných prohlížečích udržela jejich zamýšlená podoba. Bylo ukázáno několik možných způsobů řešení problémů nestejné interpretace kaskádových stylů, je však třeba mít na zřeteli, že princip řešení využívající jedné chyby prohlížeče k vyřešení chyby jiné je poněkud sporný, protože nelze odhadnout, které chyby budou v následujících verzích prohlížeče odstraněny a které nikoliv.

Je na první pohled patrné, že většina problémů se týká Internet Exploreru, vzhledem ke stále majoritnímu zastoupení tohoto prohlížeče však nelze tyto problémy přehlížet. Na druhou stranu zastoupení a význam Internet Exploreru není natolik dominantní, aby pro tvůrce stránek bylo přijatelné optimalizovat stránky pouze pro tento prohlížeč, snad s výjimkou některých firemních intranetových aplikací, což nesouvisí pouze s CSS, ale i dalšími specifiky MSIE.

Ke konci roku 2006 se na trh dostává Internet Exploreru 7, ve kterém již je opravena většina závažných chyb předchozích verzí, několik bugů ale v prohlížeči zůstalo.

Co se týká porovnávání zbylých dvou prohlížečů, Mozilly a Opery, je implementace CSS na takové úrovni, že se výběr uživatele řídí jinými aspekty, např. uživatelským rozhraním a ovládáním.

Bohužel je třeba konstatovat, že v blízké budoucnosti nelze očekávat, že by se problémy s rozdílnou interpretací kaskádových stylů a nutnost jejich řešení staly minulostí. Dá se však předpokládat, že co se současně nejpoužívanějšího CSS 2.1 týče, situace se bude zlepšovat. Spousta nových možností a s nimi

spousta nových problémů přijde až po schválení a prvních implementacích CSS 3.

8 . Seznam literatury

[1] World Wide Web Consortium, URL: < http://www.w3.org/Consortium/ >

[2] Webtip.cz - kaskádové styly CSS 2. díl, URL: < http://www.webtip.cz/art/wt_tech_html/wt_css02.html >

[3] Toplist.cz, Údaje ze dne 27.6.2007, URL: < http://history.toplist.cz/stat/?a=history&type=1 >

[4] BIEN, Jan. Interval.cz - Implementační chyby v prohlížečích, Publikováno 9. 1. 2004, URL: < http://interval.cz/clanky/implementacni-chyby-v-prohlizecich-chybne-rozmery-boxu/ >

[5] STANÍČEK, Petr. CSS Kaskádové styly Kompletní průvodce, Computer Press, 2003. ISBN: 80-7226-872-4

[6] CSS contents and browser compatibility , URL: < http://www.quirksmode.org/css/contents.html >

[7] XUL Planet, URL: < http://www.xulplanet.com/references/elemref/ref_StyleProperties.html >

[8] JAKEŠ, Martin. Knihovna CSS – Interval.cz, URL: < http://css.interval.cz/clanky/css-zvlast-pro-msie-a-ostatni-prohlizece/ >

[9] Imfo.ru, URL: < http://imfo.ru/csstest/css_hacks/import.php >

[10] STANÍČEK, Petr. Dílna CSS & Webdesignu, ©2004, Updated May 28, 2004 URL: < http://www.wellstyled.com/css-underscore-hack.html >

[11] ÇELIK, Tantek. Box Model Hack, URL: < http://tantek.com/CSS/Examples/boxmodelhack.html >

[12] DLOUHÝ, Vít. Dlouhý web < http://www.vitdlouhy.cz/weblog/2005/02/kouzelny-msie-bug.php >

[13] GALLANT, John, BERGEVIN, Holly. Community MX, URL: < http://www.communitymx.com/content/article.cfm?page=2&cid=C37E0 >

[14] STANÍČEK, Petr. Pixilophone, URL: < http://www.pixy.cz/pixylophone/2003_08_archiv.html#1061885333 >

[15] POLAKOVIČ, Jaroslav. IE Brouci, URL: < http://ie-brouci.dero.name/peekaboo.html >

[16] GALLANT, John, BERGEVIN, Holly. Position Is Everything, Updated: January 19, 2004, URL: < http://www.positioniseverything.net >

[17] CSS Creator Forum, URL: < http://www.csscreator.com/css-forum/ntopic2718.html&sid=494a2df12b48a1e8fdca8ae0e7bfb752 >

[18] HORVATH, Janos. URL: < http://users.hszk.bme.hu/~hj130/css/list_menu/hover/ >

[19] POLAKOVIČ, Jaroslav. Updated: 7. 9. 2006, URL: < http://dero.name/weblog/ie7-bug-mizejici-obsah-zmizik/ >

9 . Použité zdroje

9.1. Seznam obrázků

9.2. Seznam grafů

9.3. Seznam příkladů