Mark Pilgrim

Ponořme se do HTML5

Edice CZ.NIC

Mark Pilgrim
PONOŘME SE DO HTML5

Vydavatel:
CZ.NIC, z. s. p. o.
Milešovská 5, 130 00 Praha 3
Edice CZ.NIC
www.nic.cz

1. vydání, Praha 2014
Kniha vyšla jako 10. publikace v Edici CZ.NIC.
ISBN 978-80-905802-6-8

CZ.NIC | SPRÁVCE DOMÉNY CZ

ISBN 978-80-905802-6-8

— Mark Pilgrim

Ponořme se do HTML5

— Edice CZ.NIC

Předmluva vydavatele

— Předmluva vydavatele

Vážení čtenáři,

po sérii knížek zaměřených převážně na běžného uživatele Internetu se v Edici CZ.NIC opět vracíme k tématu, které bude blízké zejména vývojářům internetových aplikací. A nevybrali jsme si pro tento návrat nic menšího než jazyk HTML, jeden ze základních stavebních kamenů konceptu WWW, který byl bezpochyby hlavním strůjcem rozmachu Internetu na přelomu tisíciletí. Zdá se, že po krátkém období nejistoty, jakým směrem se vývoj HTML bude ubírat, chytá tento jazyk druhý dech. Na podzim loňského roku došlo ke standardizaci HTML5 a o této nejnovější verzi jazyka pojednává právě tato kniha. V roce, kdy HTML slaví 25. výročí své existence, je již zřejmé, že nová verze bude akceptována hlavními hráči a postupně vytlačí některé proprietární technologie, které v mezičase zaplnily uvolněný prostor.

Přestože originál knihy Ponořme se do HTML5 vznikl již v roce 2011, věříme, že poskytne čtenáři dostatečný vhled jak do historie tohoto jazyka, tak do všech jeho podstatných vlastností. Zárukou je pro nás osobnost autora Marka Pilgrima, který je autorem i jiné knihy v naší edici, a to Ponořme se do Pythonu 3. Tato kniha je v Edici CZ.NIC jednou z nejoblíbenějších, a to zejména díky čtivému stylu, kterým autor čtenáře doslova vtáhne do popisovaného tématu. Autor, který v roce 2011 trochu tajemně zmizel z veřejného internetového prostoru, neustoupil ze svého originálního stylu ani v této knize.

Přeji Vám pěkné čtení.

Jaromír Talíř, CZ.NIC
Praha, 23. června 2015

— Předmluva vydavatele

Obsah

— Obsah

Obsah

Pět věcí, které byste měli vědět o HTML5

1. Není to jedna velká věc

Možná se ptáte: „Jak můžu začít používat HTML5, pokud ho nepodporují starší prohlížeče?“ Ale ta otázka je sama o sobě zavádějící. HTML5 není jedna veličina, ale soubor jednotlivých prvků. Nemůžete detekovat „podporu HTML5“, protože to by nedávalo žádný smysl. Můžete ale detekovat podporu jednotlivých prvků, jako jsou plátno, video nebo geolokace.

HTML si lze představovat jako tagy a špičaté závorky. Ty jsou jistě jeho důležitou součástí, ale není to všechno. Specifikace HTML5 také určují, jak tyto špičaté závorky fungují s JavaScriptem prostřednictvím Document Object Model (DOM). Není to tak, že by HTML5 pouze definoval tag `<video>`, ale v DOMu se nachází také odpovídající DOM API pro videoobjekty. Toto API můžete použít pro detekci podpory jednotlivých formátů videa, přehrání videa, jeho pozastavení, vypnutí zvuku, zjištění, kolik procent videa se stáhlo, a všeho dalšího, co potřebujete, abyste uživateli spolu s tagem `<video>` poskytli dostatečný komfort.

Kapitola 2 a Příloha A vás naučí jak správně detekovat podporu pro každý z nových prvků HTML5.

2. Nemusíte nic vyhazovat

Ať už ho milujete, nebo nenávidíte, nemůžete popřít, že HTML4 je nejúspěšnější značkovací formát všech dob. Na tomto úspěchu staví HTML5. Nemusíte vyhazovat už jednou napsané kódy, ani se nemusíte znova učit věci, které už znáte. Pokud vaše webové aplikace včera běžela v HTML4, bude běžet také dnes v HTML5. Opravdu.

Ovšem pokud chcete své webové aplikace *vylepšit*, jste tady správně. Uveďme si jeden příklad: HTML5 podporuje všechny formulářové prvky z HTML4, ale také obsahuje nové vstupní prvky. Mezi ně patří doplňky, na které jsme čekali už dlouho, jako posuvníky nebo prvky pro výběr data, zatímco jiné jsou méně nápadné. Například vstupní prvek typu `email` vypadá jako obyčejné textové pole, ale mobilní prohlížeče upraví klávesnici na displeji tak, aby se daly snadněji vkládat e-mailové adresy. Starší prohlížeče, které nepodporují vstupní prvek typu `email`, s tímto prvkem budou zacházet jako s klasickým textovým polem a formulář bude fungovat i nadále bez změny kódu nebo skriptování. To znamená, že můžete začít vylepšovat svůj web už dnes, i když budou někteří vaši návštěvníci stále ještě používat IE6.

Všechny šťavnaté detaily o formulářích HTML se dozvíte v kapitole 9.

3. Začít je snadné

„Upgradovat“ na HTML5 jde i pouhou změnou *doctype*. Doctype by měl být přítomen na prvním řádku kódu každé stránky HTML. Předchozí verze HTML definovaly velké množství doctype a vybrat si ten správný může být problematické. V HTML5 je pouze jediný doctype:

```
<!DOCTYPE html>
```

Upgradování na doctype HTML nerozbije váš stávající kód, protože zastaralé prvky, které předtím definovalo HTML 4, se budou zobrazovat i nadále v HTML5. Ale umožní vám to použít – a validovat – nové sémantické prvky, jako jsou `<article>`, `<section>`, `<header>` a `<footer>`. Všechno o těchto nových prvcích se dozvíte v kapitole 3.

4. Už to funguje

Ať už chcete něco vykreslit na canvasu, přehrát video, navrhnout lepší formuláře nebo vytvořit webové aplikace, které fungují i offline, zjistíte, že HTML5 má už teď velmi dobrou podporu. Firefox, Safari, Chrome, Opera a mobilní prohlížeče už teď podporují plátno (kapitola 4), video (kapitola 5) a geolokaci (kapitola 6), místní úložiště (kapitola 7) a další. Google již nyní podporuje anotaci pomocí mikrodat (kapitola 10). Dokonce Microsoft, který těžko může někdo osočovat z průkopnictví v oblasti podpory standardů, v Internet Exploreru 9 podporuje většinu prvků HTML5.

Každá kapitola této knihy obsahuje až příliš dobře známé tabulky kompatibility prohlížečů. Ale co je podstatnější, v každé kapitole najdete také otevřenou diskuzi o možnostech, které máte, když potřebujete podporu pro starší prohlížeče. Prvky HTML5 jako geolokace (kapitola 6) a video (kapitola 5) se poprvé objevily v podobě prohlížečových zásuvných modulů typu Gears nebo Flash. Další funkce, jako je plátno (kapitola 4), lze plně emulovat v JavaScriptu. Tato kniha vás naučí jak využívat nové funkce moderních prohlížečů, aniž byste zapomínali na prohlížeče starší.

5. Už tu s námi zůstane

Tim Berners-Lee vynalezl web na začátku devadesátých let. Později založil sdružení W3C, aby fungovalo jako strážce internetových standardů, což tato organizace dělá už déle než 15 let. V červenci roku 2009 sdružení W3C o budoucnosti internetových standardů řeklo následující:

> Ředitel dnes oznámil, že až pracovní skupina XHTML 2 skončí podle plánu na konci roku 2009, nebude obnovena. Tímto krokem a navýšením zdrojů pro pracovní skupinu

HTML chce W3C urychlit vývoj HTML5 a vyjasnit postoj organizace ohledně budoucnosti HTML.

HTML5 tu s námi už zůstane. Tak pojďme na to.

1. Jak jsme se sem dostali

1. Jak jsme se sem dostali – 21

1.1 Začínáme

Nedávno jsem narazil na něco, co řekl jeden z vývojářů Mozilly ohledně napětí, které nutně vzniká při tvoření standardů:

> Implementace a specifikace spolu musejí jít ruku v ruce. Nechcete, aby implementace proběhla dřív, než jsou hotové specifikace, protože lidé začnou být závislí na detailech implementací, což omezuje specifikace. Také ovšem nechcete, aby byly specifikace hotové dřív, než vzniknou implementace a s nimi spojené zkušenosti, protože potřebujete zpětnou vazbu. Proto vzniká nevyhnutelné napětí, ale my se s ním zkrátka musíme nějak poprat.

Na tento citát myslete, když vám budu vysvětlovat, jak vzniklo HTML5.

1.2 Typy MIME

Tato kniha pojednává o HTML5, nikoliv o předchozích verzích HTML, ani o žádné verzi XHTML. Ale abychom pochopili historii HTML5 a důvody pro jeho vytvoření, musíme nejprve pochopit několik technických detailů. Konkrétně jsou to typy MIME.

Pokaždé, když váš webový prohlížeč vyšle požadavek na stránku předtím, než pošle skutečný zdrojový kód, posílá webový server tzv. hlavičku. Tyto hlavičky jsou obvykle neviditelné, ale existují nástroje pro tvorbu webových stránek, které je zviditelňují, pokud si to přejete. Ale tyto hlavičky jsou důležité, protože říkají vašemu prohlížeči, jak má interpretovat zdrojový kód stránky, který bude následovat. Nejdůležitější hlavička se jmenuje `Content-Type` a vypadá takto:

```
Content-Type: text/html
```

„`text/html`“ se nazývá „typ obsahu“ nebo „typ MIME“ stránky. Tato hlavička je jedinou věcí, která určuje, jak vypadá konkrétní zdroj a jak by měl být interpretován. Obrázky mají své vlastní typy MIME (`image/jpeg` pro formát JPEG, `image/png` pro formát PNG apod.). Soubory JavaScript mají své vlastní typy MIME stejně jako kaskádové styly. Své vlastní typy MIME má zkrátka úplně všechno. Celý Internet funguje na základě typů MIME.

Ve skutečnosti je to samozřejmě mnohem komplikovanější. První generace webových serverů (mluvím teď o serverech z roku 1993) nevysílala hlavičku `Content-Type`, protože ta tehdy ještě neexistovala (byla vymyšlena až v roce 1994). Z důvodů týkajících se kompatibility, které se datují až do roku 1993, budou některé oblíbené prohlížeče za určitých podmínek hlavičku `Content-Type` ignorovat (říká se tomu „content sniffing“, tedy „očichání obsahu“). Ale obvykle platí pravidlo, že všechno, co jste si kdy na webu prohlíželi – HTML stránky, obrázky, skripty,

videa, dokumenty PDF, všechno s URL adresou – vám bylo naservírováno s konkrétním typem MIME v hlavičce `Content-Type`.

Zapište si to za uši, později se k tomu vrátíme.

1.3 Delší odbočka na téma tvoření standardů

Proč vlastně máme prvek `<img>`? To asi není otázka, kterou byste slyšeli každý den. Je jasné, že ho *někdo* musel vytvořit. Sám od sebe asi nevznikl. Každý prvek, každý atribut, každá funkce HTML je výtvorem někoho, kdo rozhodl, jak budou fungovat, a všechno to zapsal. Tito lidé nejsou bohové, ani nejsou dokonalí. Jsou to jen lidé. Chytří lidé, to ano, ale stále jen lidé.

Jedna ze skvělých věcí na standardech, které se vyvinuly „v otevřeném prostoru" je, že se můžete vrátit v čase a na všechny takové otázky odpovědět. Diskuse se nacházejí v e-mailových konferencích, které jsou obvykle archivovány a dají se veřejně vyhledat. Proto jsem se rozhodl dát si trochu „e-mailové archeologie", abych našel odpověď na otázku „Proč máme prvek `<img>`?" Musel jsem se vrátit v čase do doby ještě předtím, než vznikla organizace nazvaná World Wide Web Consortium (W3C). Vrátil jsem se do internetového pravěku, kde jste mohli spočítat počet webových serverů na prstech obou rukou a možná ještě tak jedné nohy.

25. února roku 1993 Marc Andreessen napsal:

> Rád bych navrhl nový volitelný tag HTML:
>
> IMG
>
> Povinným parametrem je `SRC="url"`.
>
> Tímto tagem se pojmenovává bitmapový nebo pixmapový soubor, který si pak prohlížeč pokusí ze sítě stáhnout, interpretovat ho jako obrázek a vložit ho do textu v místě výskytu tagu.
>
> Příklad:
>
> ```
> <IMG SRC="file://foobar.com/foo/bar/blargh.xbm">
> ```
>
> (Zavírací tag není potřeba, jedná se o samostatný tag.)
>
> Tento tag může být zabudován do odkazu jako cokoliv jiného. V takovém případě se stane ikonou, která je citlivá na aktivaci stejně jako klasický textový odkaz.

> Prohlížeče by měly mít volnost ohledně jimi podporovaných formátů obrázků. Bylo by například dobré, aby podporovaly Xbm a Xpm. Pokud prohlížeč nedokáže daný formát přečíst, může s ním místo toho udělat cokoliv chce (v X Mosaic vyskočí jako zástupný symbol výchozí bitmapa).
>
> Tohle je požadovaná funkce pro X Mosaic. Funguje nám to a budeme to používat minimálně interně. Určitě se nebudu bránit nápadům, jak by se to mělo zvládnout v HTML. Pokud máte nějaký lepší nápad než to, co tu teď předkládám, dejte mi prosím vědět. Vím, že je to neurčité, pokud jde o formát obrázku, ale nevidím jinou alternativu než si říct „nechte prohlížeč dělat, co umí" a počkat si na to, až někdo přijde s dokonalým řešením (MIME, snad jednou v budoucnu).

Xbm a Xpm byly oblíbené grafické formáty používané v operačním systému Unix.

„Mosaic" byl jeden z nejstarších webových prohlížečů („X Mosaic" byla jeho verze, která běžela pod Unixem). Marc Andreessen napsal tuto zprávu počátkem roku 1993, tedy ještě předtím, než založil firmu, která ho proslavila – „Mosaic Communications Corporation" – a začal pracovat na vlajkové lodi této společnosti, „Mosaic Netscape" (asi je znáte lépe pod jejich pozdějšími názvy „Netscape Corporation" a „Netscape Navigator").

„MIME, snad jednou v budoucnu" je odkazem na vyjednávání obsahu (content negotiation), funkci HTTP, kde klient (např. webový prohlížeč) sdělí serveru (např. webovému serveru), jaké typy zdrojů podporuje (např. `image/jpeg`), takže server může vrátit informaci ve formátu upřednostňovaném klientem. Původní protokol HTTP, tak jak byl definován v roce 1991 (jehož jediná verze byla realizována v únoru 1993), neumožňoval klientům žádným způsobem serverům sdělit, jaké grafické formáty podporují, a proto čelil Marc Andreessen takovému designovému dilematu.

O několik hodin později Tony Johnson odpověděl:

> Mám něco hodně podobného v Midas 2.0 (ten používáme u nás v SLAC a chystáme se ho zveřejnit co nejdříve), ovšem všechny názvy se odlišují, a má to parametr navíc: `NAME="name"`. A funguje to téměř úplně stejně jako tebou navržený tag `IMG`, např.:
>
> ```
> <ICON name="NoEntry" href="http://note/foo/bar/NoEntry.xbm">
> ```
>
> Smyslem parametru name (název) je umožnit prohlížeči, aby měl sadu „zabudovaných" obrázků. Pokud se název shoduje se „zabudovaným" obrázkem, bude použit tento obrázek, aniž by bylo potřeba obrázek brát z vnějšího zdroje. Název by také mohl „line mode" (textovým) prohlížečům napovědět, jaký symbol umístit namísto obrázku.

> Je mi celkem jedno, jak se budou parametry a tagy jmenovat, ale bylo by rozumné, kdybychom používali totéž. Nejsem moc fanda zkratek, takže proč nepoužít `IMAGE=` a `SOURCE=`? Více se mi líbí `ICON`, protože to naznačuje, že by měl `IMAGE` být spíše menší, ale možná je `ICON` už příliš významově zatížené slovo?

Midas byl další ze starých webových prohlížečů, současník „X Mosaic“. Tento multiplatformní prohlížeč běžel pod Unixem a VMS. SLAC je zkratka pro Stanford Linear Accelerator Center, nyní SLAC National Accelerator Laboratory, centrum, které provozovalo první webový server ve Spojených státech (a zároveň první webový server mimo Evropu). Když Tony Johnson psal tuto zprávu, byl SLAC na poli WWW už zkušeným harcovníkem, jelikož na svém webovém serveru hostoval pět stránek po celých 441 dní.

Tony pokračoval:

> Když už mluvíme o nových tazích, mám další, v některých ohledech podobný tag, pro který bych chtěl podporu v Midas 2.0. V zásadě jde o:
>
> ```
> <INCLUDE HREF="...">
> ```
>
> Záměrem je, aby byl druhý dokument obsažen v prvním dokumentu v místě, kde se vyskytuje tag. V zásadě může být odkazovaným dokumentem cokoliv, ale hlavním účelem je umožnit, aby byly do dokumentů vkládány obrázky (v tomto případě libovolně velké). S příchodem HTTP2 by se pak o formátu vkládaného dokumentu zvlášť jednalo.

„HTTP2“ je odkazem na Základní protokol HTTP, jak byl definován v roce 1992. V tomto okamžiku, tj. na počátku roku 1993, byl stále z větší části nezrealizován. Návrh známý jako „HTTP2“ byl dále vyvíjen a nakonec byl standardizován jako „HTTP 1.0“ (ovšem až za další tři roky). HTTP 1.0 už zahrnoval hlavičky požadavku pro vyjednání obsahu neboli „MIME, snad jednou v budoucnu.“

Tony pokračoval:

> Jako alternativa mě napadlo:
>
> ```
> <A HREF="..." INCLUDE>Zobrazit fotografii</A>
> ```
>
> Příliš se mi nelíbí přidávat další funkce do tagu `<A>`, ale cílem je zde zachovat kompatibilitu i pro prohlížeče, které si neporadí s parametrem `INCLUDE`. Záměrem je, aby prohlížeče, které rozumí `INCLUDE`, nahradily text odkazu (v tomto případě „Zobrazit fotografii“) vloženým dokumentem (obrázkem), zatímco starší nebo „hloupější“ prohlížeče budou tag `INCLUDE` zcela ignorovat.

Tento návrh nebyl nikdy uskutečněn, ale myšlenka nahrazení obrázku textem, pokud obrázek chybí, je důležitou technikou pro zvýšení přístupnosti, která v Marcově původním návrhu `<IMG>` chyběla. O několik let později byla tato funkce přidána jako atribut `<img alt>`, což Nescape okamžitě pokazil tím, že s ním omylem zacházel jako s popiskem.

Za několik hodin poté, co Tony odeslal svou zprávu, odpověděl Tim Berners-Lee:

> Představoval jsem si, že obrázky budou znázorněny jako:
>
> ```
> <a name="fig1" href="fghjkdfghj" REL="EMBED, PRESENT">Obrázek</a>
> ```
>
> kde relační hodnoty znamenají:
>
> ```
> EMBED     Vložte zde při předložení
> PRESENT   Předložte vždy, když je předložen zdrojový dokument
> ```
>
> Vezměte na vědomí, že můžete použít různé kombinace těchto hodnot, a pokud prohlížeč nepodporuje žádný z nich, kód se nerozbije.
>
> Chápu, že když se to použije jako metoda pro volitelné ikony, znamená to vkládání odkazů do sebe. Hmm. Ale speciální tag jsem nechtěl.

Tento návrh nebyl nikdy zrealizován, ale s atributem `rel` se můžeme setkat dodnes.

Jim Davis dodal:

> Bylo by fajn, kdyby se dal nějak specifikovat typ obsahu, např.:
>
> ```
> <IMG HREF="http://nsa.gov/pub/sounds/gorby.au" CONTENT-TYPE=audio/
> basic>
> ```
>
> Ale jsem zcela ochoten se podřídit požadavku, abych určoval typ obsahu příponou souboru.

Ani tento návrh nebyl nikdy zrealizován, ale Netscape později přidal podporu vkládání mediálních objektů pomocí prvku `<embed>`.

Jay C. Weber se zeptal:

> I když mému žebříčku typů médií, u kterých bych chtěl, aby je podporoval WWW prohlížeč, vévodí obrázky, nemyslím si, že bychom měli vymýšlet speciální techniky pro

> každé médium zvlášť. Co se stalo s nadšením pro používání mechanismu založeného na typech MIME?

Marc Andreessen odpověděl:

> Toto není náhrada za blížící se používání MIME jako standardního dokumentového mechanismu. Jedná se o nezbytnou a jednoduchou implementaci funkce, která je potřebná nezávisle na MIME.

Jay C. Weber odpověděl:

> Pojďme na chvíli zapomenout na MIME, aby nás to nepletlo. Moje námitka byla vedena proti diskuzi na téma „jak budeme podporovat vložené obrázky" spíše než na téma „jak budeme podporovat vložené objekty v různých médiích".
>
> Jinak příští týden někdo přijde s návrhem na nový tag `<AUD SRC="file://foobar.com/foo/bar/blargh.snd">` pro audio.
>
> Používat něco, co se dá uplatnit obecněji, by neměl být takový problém.

Když se ohlédneme nazpět, vypadá to, že Jayovy obavy byly zcela opodstatněné. Trvalo to o trochu víc než týden, ale HTML5 konečně přidalo nové prvky `<video>` a `<audio>`.

V odpovědi na Jayovu původní zprávu Dave Raggett řekl:

> Přesně tak! Chci uvážit celou řadu možných typů obrázků/čárové grafiky, spolu s možností vyjednávání formátu. Timova poznámka o podpoře oblastí uvnitř obrázků s možností kliknutí je také důležitá.

O něco později v roce 1993 Dave Raggett navrhl HTML+ jako evoluci standardu HTML. Tento návrh nebyl nikdy realizován a byl překonán HTML 2.0. HTML 2.0 bylo „retrospektivní", což znamená, že formalizovalo prvky, které se už v té době běžně používaly. „Tato specifikace spojuje, vyjasňuje a formalizuje řadu prvků, které zhruba odpovídají schopnostem HTML, jak se běžně používalo do června 1994."

Dave později napsal HTML 3.0 založené na jeho předchozím návrhu HTML+. HTML 3.0 nebylo využito nikde kromě Areny, vlastního prohlížeče W3C, a bylo překonáno HTML 3.2, které bylo opět „retrospektivní". „HTML 3.2 přidává prvky s širokou škálou využití, jako jsou tabulky, aplety a tok textu kolem obrázku, a zároveň poskytuje plnou zpětnou kompatibilitu s existujícím standardem HTML 2.0."

Později se Dave stal spoluautorem HTML 4.0, vyvinul HTML Tidy a podílel se i na XHTML, XForms, MathML a dalších moderních specifikacích W3C.

Vraťme se do roku 1993, kdy Mark odpověděl Daveovi:

> Možná bychom se vlastně měli zamyslet nad univerzálním procedurálním grafickým jazykem, ve kterém bychom mohli vkládat libovolné hypertextové odkazy připojené k ikonám, obrázkům nebo k textu, k čemukoliv. Má někdo představu, co v tomhle ohledu dokáže Intermedia?

Intermedia byl hypertextový projekt Brownovy univerzity. Byl vyvíjen mezi lety 1985 a 1991 a běžel pod A/UX, což byl operační systém podobný Unixu určený pro starší typy počítačů Macintosh.

Myšlenka „univerzálního procedurálního grafického jazyka“ se nakonec uchytila. Moderní prohlížeče podporují jak SVG (značkovací jazyk s vloženými skripty), tak `<canvas>` (procedurální grafický API s přímým vykonáváním instrukcí), ovšem `<canvas>` se původně objevil jako proprietární rozšíření, než se stal „retrospektivním“ díky WHATWG.

Bill Janssen odpověděl:

> Další systémy, jimiž se můžeme inspirovat, které obsahují tento (celkem cenný) koncept, jsou Andrew a Slate. Andrew je postavený pomocí _insetů_, z nichž každý má nějaký zajímavý typ, jako je text, bitmapa, kresba, animace, zpráva, tabulka atd. Funguje zde koncept arbitrárního rekurzivního vkládání, takže jakýkoliv inset jakéhokoliv druhu může být vložen do jakéhokoliv jiného druhu, který podporuje vkládání. Například může být inset vložen do libovolného místa textu v textovém widgetu nebo do jakékoliv obdélníkové oblasti v kreslícím widgetu či do libovolné buňky v tabulce.

„Andrew“ odkazuje na Andrew User Interface System (v té době ovšem známý prostě jenom jako Andrew Project).

Mezitím dostal Thomas Fine jiný nápad:

> Podle mě je nejlepším způsobem, jak zacházet s obrázky ve WWW, používání MIME. Jsem si jistý, že postscript je už v tuto chvíli v MIME podporovaný podtyp, a právě on si dokáže velmi dobře poradit s mícháním textu a grafiky.
>
> Namítáte, že se na něj nedá klikat? Ano, máte pravdu. Myslím, že s tímhle si už teď dokáže poradit display poscript. A i kdyby náhodou ne, tak doplnění standardního postscriptu je jednoduchá záležitost. Definujte příkaz s odkazem, který specifikuje URL, a použije

> současnou cestu jako uzavřenou oblast pro tlačítko. Jelikož to postscript tak dobře umí s cestami, budou libovolné tvary tlačítka brnkačka.

Display Postscript byla zobrazovací technologie pro obrazovkové servery, na jejímž vývoji se podílely firmy Adobe a NeXT.

Tento návrh nebyl nikdy zrealizován, ale myšlenka, že nejlepším způsobem jak vylepšit HTML je nahradit ho něčím úplně jiným, se stále čas od času objevuje.

Tim Berners-Lee, 2. března 1993:

> HTTP2 umožňuje dokumentu, aby obsahoval jakýkoliv typ, o kterém uživatel řekl, že ho zvládne, a nikoliv pouze registrované typy MIME. Proto můžeme experimentovat. Ano, myslím, že existuje dost argumentů pro spojení postscriptu s hypertextem. Nevím, zda stačí Display Postcsript. Vím, že se Adobe snaží vytvořit svoje vlastní na postcriptu založené „PDF", které bude čitelné v jejich vlastním druhu prohlížecích programů.
>
> Myslel jsem, že obecný zastřešovací jazyk pro odkazy (založený na Hytime?) by umožnil standardům pro hypertext a grafiku/video se vyvíjet samostatně, což by pomohlo obojímu.
>
> Místo tagu `IMG` použijme `INCLUDE` a nechme ho odkazovat k libovolnému typu dokumentu. Nebo použijme `EMBED`, pokud `INCLUDE` zní jako příkaz v C++, od kterého by lidé čekali, že poskytne zdrojový kód SGML, aby byl analyzován inline – což rozhodně nebyl záměr.

HyTime byl raný, na SGML založený hypertextový dokumentový systém. Často se přetřásal v časných diskuzích na téma HTML a později XML.

Timův návrh na tag `<INCLUDE>` nebyl nikdy realizován, i když jeho ozvěny můžeme zaznamenat v prvcích `<object>`, `<embed>` a `<iframe>`.

Konečně 12. března 1993 se Marc Andreessen vrátil k tomuto diskuznímu vláknu:

> Zpátky k vláknu o vkládání obrázků – už brzo vydám Mosaic v 0.10, která bude podporovat vkládání GIFů a XMB obrázků/bitmap, jak jsem psal dříve. (…)
>
> V tuto chvíli nejsme připraveni na podporu `INCLUDE/EMBED`. (…) Takže nejspíš použijeme `<IMG SRC="url">` (nikoliv `ICON`, protože ne všechny vkládané obrázky je tak možné nazývat). Prozatím u vkládaných obrázků nebude uveden content-type; do budoucna plánujeme jeho podporu (spolu s obecným přijetím MIME). Rutiny pro čtení obrázků,

které teď používáme, ve skutečnosti dokážou poznat formát obrázku za pochodu, takže na příponě souboru nezáleží.

1.4 Nepřerušená linie

Všechny aspekty této téměř 19 let staré konverzace vedoucí ke vzniku prvku HTML, který se používá snad na každé webové stránce, jež kdy byla vytvořena, mě nesmírně fascinují. Uvažte následující:

- Protokol HTTP stále existuje. HTTP se úspěšně vyvinul z 0.9 do 1.0 a později 1.1. A vyvíjí se dál.

- HTML stále existuje. Ten primitivní datový formát – který ani nepodporoval vložené obrázky! – se úspěšně vyvinul do verzí 2.0, 3.2, 4.0. HTML tvoří nepřerušenou linii. Určitě je to linie pokroucená, zauzlovaná a zasukovaná. Na vývojovém stromě HTML bylo spousta „uschlých ratolestí", míst, kde se lidé, kteří měli standardy na starosti, příliš rozjeli (a to na úkor autorů a realizátorů). Ale i přesto. Jsme v roce 2012 a webové stránky z roku 1990 se stále zobrazují v moderních prohlížečích. Právě jsem jednu načetl ve svém nejnovějším mobilu s Androidem, a ani mi přitom nevyskočilo upozornění „prosím čekejte, než se převede historický formát…".

- HTML bylo vždy konverzací mezi tvůrci prohlížečů, autory stránek, tvůrci standardů a dalšími lidmi, kteří se vynořili bůhví odkud, aby si povídali o špičatých závorkách. Většina úspěšných verzí HTML byla „retrospektivní" – dohánělа technický svět, a přitom se ho snažila pošoupnout správným směrem. Každý, kdo vám říká, že by mělo HTML zůstat „čisté" (nejspíš ignorováním tvůrců prohlížečů nebo ignorováním autorů, případně obojím), je jednoduše vedle. HTML nebylo nikdy čisté, a všechny pokusy ho očistit byly natolik katastrofální, že se jim vyrovnaly pouze pokusy HTML něčím nahradit.

- Žádný z prohlížečů z roku 1993 neexistuje v podobě, v jaké bychom ho mohli poznat. Nestcape Navigator byl opuštěn v roce 1998 a od základů přepsán, aby vznikla Mozilla Suite, od které se pak odtrhl Firefox. Internet Explorer si prošel skromnými začátky v „Microsoft Plus! pro Windows 95", kde se ocitl v jednom balíčku s motivy plochy a pinballem. (Ale samozřejmě po jeho stopách můžeme pátrat také v ještě vzdálenější minulosti.)

- Některé z operačních systémů z roku 1993 stále existují, ale žádné z nich nemají vztah k modernímu Internetu. Většina lidí, kteří chodí na web, k tomu využívají PC s operačním systémem Windows 2000 nebo s pozdější verzí, Mac s Mac OS X, PC s určitou odnoží Linuxu nebo kapesní zařízení jako iPhone. V roce 1993 Windows běžel pod verzí

3.1 (a soutěžil s OS/2), Macy používaly System 7 a Linux se šířil přes Usenet. (Chcete si užít trochu legrace? Najděte nějakého veterána a pošeptejte mu „Trumpet Winsock" nebo „MackPPP.")

- Někteří z těchto lidí stále pracují v oboru a podílejí se na tom, čemu teď říkáme jednoduše „webové standardy". A to i po téměř 20 letech. A někteří z nich se podíleli už na předchůdcích HTML, s datací do 80. let i dříve.

- Když je řeč o předchůdcích… Díky výsledné popularitě HTML a webu je snadné zapomínat na další tehdejší formáty a systémy, které ovlivnily jejich výslednou podobu. Andrew? Intermedia? HyTime? HyTime přitom nebyl žádný obskurní akademický projekt – byla to norma ISO. Schválili ho pro vojenské použití. Zkrátka důležitá záležitost. Ostatně si o něm můžete přečíst sami… na téhle stránce, HTML ve svém webovém prohlížeči[1].

Ale nic z toho neodpovídá na původní otázku: proč máme prvek `<img>`? Proč ne prvek `<icon>`? Nebo prvek `<include>`? Proč nemáme hypertextový odkaz s atributem `<include>` nebo nějakou kombinaci hodnot `rel`? Proč zrovna prvek `<img>`? Docela jednoduše proto, že Marc Andreessen svůj kód oficiálně vydal, a vydané kódy vyhrávají.

To samozřejmě neznamená, že *všechny* vydané kódy vyhrávají; Andrew, Intermedia a HyTime své kódy přece také vydaly. Kód je nezbytnou podmínkou pro úspěch, ale sám o sobě nestačí. A už *vůbec* nechci říct, že vydání kódu před vznikem standardu je nejlepší řešení. Marcův prvek `<img>` nestanovil společný grafický formát, neurčoval, jak bude kolem něho proudit text, a nepodporoval textové alternativy ani nouzový obsah pro starší prohlížeče. A o sedmnáct let později stále ještě bojujeme s očicháváním obsahu a je to stále zdrojem neuvěřitelných bezpečnostních děr. A tohle všechno můžete vysledovat do doby před 17 lety, napříč velkými válkami prohlížečů, k 25. únoru 1993, kdy Marc Andreessen jen tak mimoděk prohodil „MIME, snad jednou v budoucnu," a pak svůj kód stejně vydal.

Ti, kdo vyhrávají, jsou ti, kdo vydávají.

1.5 Historie vývoje HTML mezi lety 1997 a 2004

V prosinci 1997 vydalo sdružení World Wide Web Consortium (W3C) HTML 4.0 a okamžitě ukončilo pracovní skupinu HTML. Ani ne o dva měsíce později jiná pracovní skupina W3C vydala XML 1.0. Pouhé tři měsíce nato provozovatelé W3C uspořádali workshop na téma „Utváření budoucnosti HTML", aby odpověděli na otázku: „Vykašlala se W3C na HTML?"

1 http://www.sgmlsource.com/history/hthist.htm

Tohle byla jejich odpověď:

> Diskuze dospěly k názoru, že další rozšiřování HTML 4.0 by bylo obtížné stejně jako převod 4.0 na aplikaci XML. Navrhovaný způsob, jak se lze vypořádat s těmito omezeními, je začít znovu s novou generací HTML založenou na balíku sad tagů XML.

Sdružení W3C obnovilo pracovní skupinu HTML a pověřilo ji vytvořením tohoto „balíku sad tagů XML". Prvním krokem byl v prosinci roku 1998 návrh prozatímní specifikace, která jednoduše převedla HTML do XML bez přidávání nových prvků či atributů. Tato specifikace se později stala známou pod názvem „XHTML 1.0". Definovala nový typ MIME pro dokumenty XHTML `application/xhtml+xml`. Pro usnadnění přechodu pro existující stránky HTML4 byl také zahrnut Dodatek C, který „shrnuje pokyny pro tvorbu stránek pro autory, kteří chtějí, aby se jejich dokumenty XHTML zobrazovaly v existujících uživatelských agentech HTML." Dodatek C říkal, že jste mohli tvořit takzvané „XHTML stránky", ale vydávat je přitom s typem MIME `text/html`.

Jejich dalším cílem byly webové formuláře. V roce 1999 tatáž pracovní skupina HTML vydala první návrh „Rozšířených formulářů XHTML". Svá očekávání popsala v prvním odstavci:

> Po pečlivém zvážení se pracovní skupina HTML rozhodla, že cíle pro další generaci formulářů jsou neslučitelné se zachováním zpětné kompatibility s prohlížeči vytvořenými pro dřívější verze HTML. Naším cílem je vytvořit nový čistý model formuláře („Rozšířené formuláře XHTML") založený na jasně definovaných požadavcích. Požadavky popsané v tomto dokumentu se zakládají na zkušenostech s celou škálou formulářových aplikací.

O několik měsíců později byly „Rozšířené formuláře XHTML" přejmenovány na „XForms" a dostaly svoji vlastní pracovní skupinu. Tato skupina pracovala paralelně s pracovní skupinou HTML a konečně vydala první verzi XForms 1.0 v říjnu 2003.

Mezitím se po dokončení přechodu na XML pracovní skupina HTML zaměřila na vytvoření „nové generace HTML". V květnu 2001 vydali první verzi XHTML 1.1, která přidávala k XHTML jen několik málo prvků navíc, ale také rušila zadní vrátka v Dodatku C. Od verze 1.1 měly všechny dokumenty XHTML obsahovat typ MIME `application/xhtml+xml`.

1.6 Všechno, co víte o XHTML, je špatně

Proč jsou typy MIME důležité? Proč se k nim neustále vracím? Jen tři slova: „draconian error handling", tj. přísné zacházení s chybami. Prohlížeče se vůči HTML vždycky chovaly benevolentně. Pokud vytvoříte HTML stránku a zapomenete na tag `</head>`, prohlížeče stránku i přesto zobrazí. (Některé tagy implicitně vyvolají konec `<head>` a začátek `<body>`.) Tagy byste

měli uzavírat hierarchicky, to znamená od posledního k prvnímu, ale když napíšete kód ve stylu `<b><i></b></i>`, prohlížeče si s tím (nějak) poradí a zobrazí stránku bez chybového hlášení.

Jak se dá očekávat, skutečnost, že „vadné" HTML stránky v prohlížečích fungovaly, vedla autory k vytváření vadných HTML stránek. Velké spousty vadných stránek. Podle některých odhadů má až 99 % všech HTML stránek na Internetu alespoň jednu chybu. Ale protože na tyto chyby prohlížeče nevyhazují žádná viditelná chybová hlášení, nikdo je nikdy neopraví.

W3C tohle vnímalo jako základní problém Internetu a rozhodlo se to napravit. XML vydané v roce 1997 porušilo tradici benevolentních klientů, když stanovilo podmínku, že všechny programy, které konzumují XML, musí zacházet s tzv. „well-formedness" chybami (chyby nesprávně strukturovaného kódu) jako s fatálními. Konceptu selhání u první chyby se začalo říkat „drakonické trestání chyb" podle starořeckého státníka Drakóna, který zavedl trest smrti i za relativně mírné porušení jeho zákonů. Když v W3C přeformulovali HTML jako slovník XML, zavedli podmínku, že všechny dokumenty vytvářené s novým typem MIME `application/xhtml+xml` budou podléhat drakonickému trestání chyb. Pokud by vaše XHTML stránka obsahovala byť jen jednu chybu v struktuře kódu – např. opomenutí tagu `</head>` nebo uzavírání tagů ve špatném pořadí – nebudou mít webové prohlížeče jinou možnost než přestat stránku zpracovávat a zobrazit koncovému uživateli chybové hlášení.

Tato myšlenka si nezískala všeobecnou oblibu. S ohledem na odhadovaných 99 % chyb v existujících stránkách a všudypřítomnou hrozbu, že se koncovému uživateli bude zobrazovat chyba, a zároveň ovšem nedostatek nových funkcí v XHTML 1.0 a 1.1, které by takové riziko vyvážily, tvůrci webů `application/xhtml+xml` v podstatě ignorovali. To ale neznamená, že by úplně ignorovali XHTML. To rozhodně ne. Dodatek C ve specifikaci XHTML 1.0 poskytl autorům webů zadní vrátka: „Použijte něco, co vypadá podobně jako syntax XHTML, ale zvolte typ MIME `text/html`." A to je přesně to, co udělaly tisíce vývojářů webových stránek – „upgradovali" na syntax XHTML, ale používali typ MIME `text/html`.

I dnes o sobě miliony webových stránek tvrdí, že jsou XHTML. Začínají s doctypem XHTML na prvním řádku, používají malá písmena pro názvy tagů a uvozovky kolem hodnot atributů a přidávají koncové lomítko po prázdných prvcích jako `<br />` a `<hr />`. Ale pouze nepatrná část těchto stránek používá typ MIME `application/xhtml+xml`, který by spustil drakonické trestání chyb XML. Jakákoliv stránka využívající typ MIME `text/html` – bez ohledu na doctype, syntax a styl kódování – bude analyzována za použití „benevolentního" analyzátoru HTML, který bude tiše ignorovat případné chyby v kódu a nikdy nebude upozorňovat koncové uživatele (ani kohokoliv jiného), i když je stránka z technického hlediska vadná.

Verze XHTML 1.0 v sobě měla tato zadní vrátka, ale verze XHTML 1.1 je zavřela, a nikdy nedokončená verze XHTML 2.0 pokračovala v tradici vyžadování drakonického trestání chyb. A to je důvod, proč máme miliony stránek, které o sobě tvrdí, že jsou XHTML 1.0, a jenom

pár, které se prohlašují za XHTML 1.1 (nebo XHTML 2.0). Je to, co používáte, opravdu XHTML? Zkontrolujte svůj typ MIME. (Pokud nevíte, jaký typ MIME používáte, můžu vám zaručit, že stále používáte `text/html`). Pokud na svých stránkách nepoužíváte `application/xhtml+xml`, vaše takzvané XHTML je XML pouze podle jména.

1.7 Konkurenční vize

V červnu 2004 pořádalo sdružení W3C workshop o webových aplikacích a složených dokumentech. Tohoto workshopu se zúčastnili zástupci tří prodejců prohlížečů, firmy zabývající se tvořením webů a další členové W3C. Skupina zúčastněných stran, včetně Mozilly Foundation a Opery Software, vystoupila s prezentací konkurenční vize budoucnosti webu: evoluce stávajícího standardu HTML 4, aby zahrnul nové funkce pro vývojáře moderních webových aplikací.

Následujících sedm principů představuje podle nás nejdůležitější požadavky pro tuto práci.

Zpětná kompatibilita, volná cesta k přechodu
: Webové aplikace by měly být založeny na technologiích, které tvůrci stránek znají, jako jsou HTML, CSS, DOM a JavaScript.

  Základní prvky webových aplikací by měly být v dnešním IE6 realizovatelné pomocí vzorců chování, skriptů a stylopisů (stylesheets), aby měli autoři volnou cestu k přechodu. Žádné řešení, které nemůže být použito s některým z uživatelských agentů, které mají v současné době vysoký podíl na trhu, bez nutnosti používání binárních zásuvných modulů, nemá příliš velkou šanci na úspěch.

Dobře definované zacházení s chybami
: Zacházení s chybami ve webových aplikacích musí být definováno do té míry, aby uživatelské agenty nemusely vytvářet své vlastní mechanismy zacházení s chybami nebo zpětně analyzovat mechanismy zacházení s chybami jiných uživatelských agentů.

Uživatelé by neměli být vystaveni chybám autorů
: Specifikace musí určit přesný způsob nápravy chyb ve všech možných chybových scénářích. Zacházení s chybami by mělo z velké části spočívat v hladké nápravě chyb (jako v CSS) spíše než v katastrofálním a do očí bijícím selhání (jako v XML).

Praktické využití
: Každý prvek ve specifikacích webových aplikací musí být odůvodněn případem praktického využití. Naopak to vždy platit nemusí: každý případ nutně nevyžaduje nový prvek.

  Případy použití by v ideálním případě měly být založeny na skutečných webech, kde autoři předtím použili nějaké špatné řešení toho, jak se poprat s nějakým omezením.

Skriptování se jen tak nezbavíme
: Ale měli bychom se mu vyhnout tam, kde může být použit praktičtější značkovací jazyk. Skriptování by mělo být nezávislé na zařízení a prezentaci, pokud není přizpůsobeno konkrétnímu zařízení (např. pokud není obsaženo v XBL).

Je třeba se vyhnout profilování pro konkrétní zařízení
: Autoři by měli mít možnost spoléhat na to, že do desktopové i mobilní verze stejného UA budou implementovány stejné funkce.

Otevřený proces
: Internet těžil z toho, že se vyvíjel v otevřeném prostředí. Webové aplikace budou tvořit jádro Internetu a jejich vývoj by se měl také odehrávat otevřeně. E-mailové konference, archivy a specifikace návrhů by měly být neustále přístupné veřejnosti.

Mezi účastníky workshopu proběhla anketa na téma „Má W3C vyvíjet deklarativní rozšíření HTML a CSS a imperativní rozšíření DOM, aby odpověděla na středně náročné požadavky webových aplikací, jako protiváhu k sofistikovaným, plně rozvinutým API na úrovni operačních systémů?“ (Anketu navrhl Ian Hickson z Opery Software.) Hlasování dopadlo 11 ku 8. Ve shrnutí průběhu workshopu sdružení W3C napsalo: „V současné době se W3C nechystá vynaložit žádné zdroje na téma třetí ankety: rozšíření HTML a CSS pro webové aplikace, kromě technologií, které jsou vyvíjeny v rámci současných pracovních skupin W3C.“

Tváří v tvář tomuto rozhodnutí měli lidé, kteří navrhovali vyvíjení HTML a formulářů HTML, pouze dvě možnosti: vzdát to, nebo pokračovat ve své práci mimo W3C. Vybrali si druhou možnost a zaregistrovali doménu `whatwg.org`. V červnu roku 2004 se pak zrodila pracovní skupina WHAT.

1.8 Pracovní skupina WHAT?

Co je vlastně zač pracovní skupina WHAT? Nechám je, aby vám to řekli sami:

> Web Hypertext Applications Technology Working Group je volné, neoficiální a otevřené sdružení výrobců webových prohlížečů a zúčastněných stran. Tato skupina usiluje o vývoj specifikací založených na HTML a souvisejících technologiích, aby se usnadnilo nasazení interoperabilních webových aplikací, s úmyslem předkládání výsledků normalizační organizaci. Toto předkládání by pak tvořilo základ práce na oficiálním rozšíření HTML v oblasti standardů.
>
> Vytvoření tohoto fóra následovalo po několika měsících práce na specifikacích pro takové technologie, která probíhala prostřednictvím e-mailu. Hlavním cílem až do tohoto bodu bylo rozšiřování Formulářů HTML4, aby podporovaly funkce požadované autory webů, bez porušení zpětné kompatibility u stávajícího obsahu. Tato skupina byla vytvořena, aby zajistila, že další vývoj těchto specifikací bude probíhat naprosto otevřeně, prostřednictvím veřejně archivované a přístupné e-mailové konference.

Klíčovou frází je zde „bez porušení zpětné kompatibility“. XHTML (bez zadních vrátek v podobě Dodatku C) není zpětně kompatibilní s HTML. Vyžaduje úplně nový typ MIME a vynucuje si drakonické trestání chyb pro veškerý obsah s tímto typem MIME. XForms nejsou zpětně kompatibilní s formuláři HTML, protože můžou být použity pouze v dokumentech, které používají nový typ MIME XHTML, což znamená, že si XForms rovněž vynucují drakonické trestání chyb. Všechny cesty vedou k MIME.

Místo toho, aby zahodila více než dekádu investování do HTML a způsobila, že by se 99 % existujících webových stránek stalo nepoužitelnými, se pracovní skupina WHAT rozhodla pro jiný přístup: zdokumentovala ty „benevolentní“ algoritmy pro zacházení s chybami, které prohlížeče v praxi používaly. Webové prohlížeče odjakživa ignorovaly chyby HTML, ale nikdo se nikdy neobtěžoval zapsat, jak to vlastně dělaly. Prohlížeč NCSA Mosaic měl své vlastní algoritmy pro nakládání s vadnými stránkami a Netscape se jim pokoušel přizpůsobit. Opera a Firefox se pokoušely přizpůsobit Internet Exploreru. Pak se Safari pokusil přizpůsobit Firefoxu. A tak to šlo dále a dále až dodnes. Vývojáři strávili tisíce hodin tím, že se snažili, aby jejich produkty byly kompatibilní s produkty konkurence.

Pokud vám to přijde jako ohromná spousta práce, tak to máte pravdu. Ohromná spousta práce to opravdu byla. Trvalo to pět let, ale (s výjimkou několika obskurních mezních případů) pracovní skupina WHAT nakonec úspěšně zdokumentovala, jak analyzovat syntax HTML způsobem, který by byl kompatibilní se stávajícím webovým obsahem. V konečném algoritmu není žádný krok, který by nařizoval, že by prohlížeče musely přestat načítat stránku a zobrazit koncovému uživateli chybové hlášení.

Mezitím co probíhalo všechno tohle reverzní inženýrství, pracovní skupina WHAT tiše pracovala i na několika dalších věcech. Jednou z nich byla specifikace, původně pojmenovaná Webové formuláře 2.0, která přidávala do formulářů HTML nové typy prvků. (Více o webových formulářích se dočtete v kapitole Forma šílenství.) Dalším projektem byla specifikace nazvaná „Webové aplikace 1.0“, která přinesla několik nových důležitých funkcí, jako vykreslovací plátno s přímým vykonáváním instrukcí a nativní podporu audia a videa bez zásuvných modulů.

1.9 Zpátky k W3C

Dva a půl roku se W3C a pracovní skupina WHAT víceméně vzájemně ignorovaly. Zatímco pracovní skupina WHAT se zaměřovala na webové formuláře a nové prvky HTML, pracovní skupina HTML W3C měla plné ruce práce s verzí XHTML 2.0. Ale v říjnu 2006 bylo jasné, že pracovní skupina WHAT nabrala na obrátkách, na rozdíl od XHTML2, které se stále ještě plácalo ve formě návrhu, přičemž ho nevyužíval žádný z vedoucích prohlížečů. V říjnu 2006 sám zakladatel W3C Tim Berners-Lee prohlásil, že W3C bude spolupracovat s pracovní skupinou WHAT na vývoji HTML.

Některé věci jsou jasnější, když se na ně díváme s několikaletým odstupem. HTML je nutné vyvíjet postupně. Pokus přimět svět, aby přešel na XML se všemi uvozovkami kolem hodnot atributů a lomítky v prázdných tazích a jmenných prostorech najednou, nefungoval. Většina veřejnosti vytvářející stránky HTML se nepohnula, z velké části proto, že si prohlížeče nestěžovaly. Některé velké komunity přešly a teď sklízejí ovoce dobře strukturovaných systémů, ale nejsou to zdaleka všechny. Je důležité postupně vylepšovat HTML a zároveň pokračovat v přechodu do dobře strukturovaného světa a získávat v tomto světě větší váhu.

V plánu je ustanovit úplně novou skupinu pro HTML. Na rozdíl od té předchozí bude pověřena postupným vylepšováním HTML a paralelně XHTML. Bude mít jiného předsedu a personální složení. Bude pracovat zároveň na HTML a XHTML. Tato skupina má velkou podporu u mnoha lidí, s kterými jsme mluvili, včetně tvůrců prohlížečů.

Skupina bude také pracovat na formulářích. To je složitá oblast, jelikož stávající formuláře a XForms jsou obojí formulářové jazyky. HTML formuláře jsou široce rozšířené a zároveň existuje i mnoho realizací a uživatelů XForms. Webforms navrhly rozumná rozšíření HTML formulářů. Podle vzoru Webforms chceme rozšířit HTML formuláře i my.

Jednou z prvních věcí, pro kterou se nově ustanovená pracovní skupina HTML W3C rozhodla, bylo přejmenovat „Webové aplikace 1.0“ na „HTML5“. No a jsme doma a můžeme začít s HTML5.

1.10 Postskriptum

V říjnu roku 2009 sdružení W3C zrušilo pracovní skupinu XHTML 2 a jako odůvodnění svého rozhodnutí vydalo následující prohlášení:

> Když jsme v březnu 2007 ohlásili pracovní skupiny HTML a XHTML 2, naznačili jsme, že budeme dále sondovat, zda je na trhu zájem o XHTML 2. Sdružení W3C si je vědomo důležitosti toho vyslat komunitě ohledně budoucnosti HTML jasný signál.
>
> Ačkoliv uznáváme hodnotu příspěvků pracovní skupiny XHTML 2 za léta její činnosti, po diskusi s účastníky se vedení W3C rozhodlo nechat pověření skupiny vypršet na konci roku 2009 a dále už ho neobnovovat.

Ti, kdo vyhrávají, jsou ti, kdo vydávají.

1.11 K dalšímu čtení

- The History of the Web[2] (Historie Internetu), starý stručný text Iana Hicksona
- HTML/History[3] (HTML/Historie) Michaela Smitha, Henriho Sivonena a dalších
- A Brief History of HTML[4] (Stručná historie HTML) Scotta Reynena

2 http://hixie.ch/commentary/web/history
3 http://www.w3.org/html/wg/wiki/History
4 http://atendesigngroup.com/blog/brief-history-of-html