Tvorba webových stránek

- co to je webová stránka (statická, dynamicky generovaná)
- serverový a klientský software pro provoz webového obsahu
- protokoly webu
- role W3C
- popis HTML (tagy párové a nepárové, příklady
- struktura HTML dokumentu
- CSS
- formuláře na webu
- klientské skriptování a DOM
- ActiveX, JavaApplet

Co to je webová stránka (statická, dynamicky generovaná)

Webová stránka:

- má svůj obsah a nese určitou informaci
- skládá se z textu, multimediálních dat (obrázky, videa, zvuky, ...) a odkazů, které umožňují přechod na další webové stránky
- může být uložena v podobě souborů na pevném disku nebo je poskytuta webovým serverem prostřednictvím počítačové sítě (LAN, Internet, ...)
- po síti je přenášena pomocí protokolu HTTP
- je psána pomocí HTML nebo XHTML (případně může být použito i XML, ale není to příliš pohodlné)
- zobrazuje se uživateli skrz webový prohlížeč

Webová stránka poskytuje své informace prostřednictvím WWW (World Wide Web). WWW je služba internetu – informační systém založený na hypertextovém modelu. Jedná se v podstatě o nejrozšířenější internetovou službu. Veškeré webové stránky jsou součástí WWW.

Webové stránky se dělí na:

- **statické** - obsahují stále stejný obsah, jsou uloženy v souborech
- **dynamické** - mění svůj obsah v čase, vytváří je program na straně webového serveru

Stránka se může měnit i přímo v prohlížeči použitím **skriptovacích jazyků**, **Javy**, **ActiveX** a dalších technologií.

Serverový a klientský software pro provoz webového obsahu

- **serverový software**
  - **webový server** - zpřístupňuje uživateli požadované webové stránky na základě jeho požadavku
    - generují dynamické webové stránky (používají další technologie - PHP, CGI, Perl, ...)
    - např. Apache, nginx, ISS (od Microsoftu vyžaduje Windows)

- **software pro dynamické generovaní webových stránek** - jedná se především o webové skriptovací jazyky - PHP, Perl, ..., ale i o externí programy, které vygenerují stránku na základě požadavku serveru

- **klientský software**
  - **webový prohlížeč** - slouží k zobrazení webových stránek na koncovém zařízení - nejčastěji monitoru počítače
    - např. Opera, Firefox, Chrome, Internet Explorer, Safari, Links (pracuje v textovém režimu), apod.

Protokoly webu

HTTP (port TCP/80)

Internetový protokol pro výměnu hypertextových dokumentů ve formátu HTML. Aktuální je verze 1.1 (definována v RFC 2616).

V současné době je používán i pro přenos dalších informací. Pomocí rozšíření MIME umí přenášet jakýkoli soubor (podobně jako e-mail), používá se společně s formátem XML pro tzv. webové služby (spouštění vzdálených aplikací) a pomocí aplikačních bran zpřístupňuje i další protokoly, jako je např. FTP nebo SMTP.

Používá (podobně jako některé jiné programy a technologie) tzv. jednotný lokátor prostředků - URL (Uniform Resource Locator). URL specifikuje jednoznačné umístění nějakého zdroje v Internetu.

Vlastnosti

- funguje způsobem **dotaz - odpověď** (uživatel pošle serveru dotaz a server mu zašle odpověď)
- **dotaz** - jedná se o čistý text
  - obsahuje označení požadovaného dokumentu, informace o schopnostech prohlížeče apod.
- **odpověď** - začíná několika řádky textu
  - tyto úvodní řádky popisují výsledek dotazu - zda se podařilo dokument najít, jakého je dokument typu, atd.
- za těmito úvodními řádky následují vlastní data dokumentu - text, HTML kód, MP3 soubor, obrázek, video-klip, apod.
- jednotlivé dotazy jsou na sobě nezávislé - svým způsobem nevýhoda - viz další bod
- jedná se o **bezstavový protokol** - neumí uchovávat stav komunikace, jednotlivé dotazy spolu nemají žádnou souvislost
  - to je nepříjemné při realizaci složitějších procesů přes HTTP - např. e-shop potřebuje uchovávat informace o totožnosti (identitě) zákazníka, o obsahu košíku, apod.
  - proto byl protokol rozšířen o tzv. HTTP Cookies - umožňují uložit serveru na počítači uživatele libovolné informace - ty lze využít pro uchování informací o stavu komunikace

První verze HTTP (verze 0.9) pochází z roku 1991. Obsahovala pouze metodu GET s jediným parametrem - názvem požadovaného dokumentu. Server jako odpověď poslal přímo požadovaný dokument bez hlavičky (`HTTP/... 200 OK...`). Případná chybová hlášení vracel server ve formě HTML dokumentu.

Verze 1.0 pochází z roku 1996. Zavádí **HTTP hlavičky**, metody `POST` a `HEAD` a pro identifikaci typu dokumentu používá MIME.

Verze 1.1 byla poprvé popsána v roce 1997 (RFC 2068), ale tento popis byl nahrazen v červnu 1999 (v RFC 2616). Zavádí možnost udržet TCP spojení (tzv. keep-alive connection) - to umožňuje zpracovat více dotazů po sobě v rámci jednoho TCP spojení. Dále přidává metody `OPTIONS`, `PUT`, `DELETE`, `TRACE` a `CONNECT`.

Příklad komunikace

Klient (např. webový prohlížeč) se připojí na server cs.wikipedia.org a zašle následující dotaz:

```
GET /wiki/Wikipedie HTTP/1.1
Host: cs.wikipedia.org
User-Agent: Opera/9.80 (Windows NT 5.1; U; cs) Presto/2.5.29 Version/10.60
Accept-Charset: UTF-8,*
```

Tím žádá server o zaslání dokumentu /wiki/Wikipedie na server cs.wikipedia.org, sděluje svou totožnost (Opera verze 10.60) a oznamuje, že podporuje kódování UTF-8. *(ve skutečnosti obsahuje dotaz takových informací mnohem více)*

Server odpoví:

```
HTTP/1.0 200 OK
Date: Fri, 15 Oct 2004 08:20:25 GMT
Server: Apache/1.3.29 (Unix) PHP/4.3.8
X-Powered-By: PHP/4.3.8
Vary: Accept-Encoding,Cookie
Cache-Control: private, s-maxage=0, max-age=0, must-revalidate
Content-Language: cs
Content-Type: text/html; charset=utf-8
```

za touto hlavičkou následuje jeden prázdný řádek (označení konce hlavičky) a následují vlastní data - v našem případě se bude jednat o HTML dokument. Hlavička obsahuje informaci o tom, že dotaz se podařil (první řádek: „200 OK“), datum a čas vyřízení dotazu, popis serveru, který odpovídá, informace o typu vráceného dokumentu (MIME typ text/html v kódování UTF-8) a další informace.

Dotazovací metody

HTTP definuje metody (příkazy), které se mají provést nad uvedeným objektem (dokumentem). Každý HTTP doraz začíná právě specifikací metody:

```
<metoda> <objekt> HTTP/<verze-http>
```

- **GET** - příkaz "vrať mi daný objekt"; nejpoužívanější metoda
  - požadavek na uvedený objekt - požadavek (dotaz) může obsahovat další případná data (proměnné prohlížeče, Cookies, Session ID, ...)
- **HEAD** - příkaz "vrať mi informace o daném objektu"
  - to samé jako GET, ale v odpovědi nejsou žádná data - pouze hlavičky obsahující informace o požadovaném objektu - velikost, typ, datum poslední změny, ...

- **POST** - odesílá uživatelské data na server
  - daty může být odeslaný vyplněný formulář, soubor, atd.
  - s objektem se pak zachází podobně jako při metodě GET
  - data může odeslat i metoda GET, ale POST je určen pro příliš velká data (víc než 512 bajtů - to je velikost požadavku GET), nebo pro situace, kdy nechceme, aby data nebyla součástí URL (adresy) - při POSTu jsou data obsažena v HTTP požadavku a ne v adrese.

- **PUT** - příkaz "nahraj data na server"
  - objekt je jméno vytvářeného souboru
  - používá se zřídka - pro nahrání dat na server se používá hlavně FTP nebo SCP/SSH
  - jsou pro to potřeba jistá oprávnění

- **DELETE** - příkaz "smaž uvedený objekt ze serveru"
  - jsou na to potřeba jistá oprávnění stejně jako u metody PUT

- **TRACE** - odešle kopii obdrženého požadavku zpět odesílateli
  - klient tak může zjistit, co na požadavku mění nebo přídávají servery, kterými požadavek prochází

- **OPTIONS** - dotaz na server, jaké podporuje metody

- **CONNECT** - spojí se s uvedeným objektem přes uvedený port
  - používá se při průchodu skrze proxy pro ustanovení kanálu SSL

HTTPS (port TCP/443)

Pro zabezpečenou komunikaci přes protokol HTTP se používá jeho zabezpečená varianta známá jako **HTTPS**. HTTPS je syntakticky identické jako HTTP, pouze přidává signalizaci prohlížeči, aby použil šifrovací metodu **SSL/TLS** k přenosu dat. SSL je pro HTTP vhodné, protože dokáže poskytnout ochranu přenosu, i když je ověřená pouze jedna strana komunikace (typicky pouze server a uživatel např. jenom potvrdí certifikát).

Druhou možností zabezpečení přenosu přes protokol HTTP je nadstavba protokolu HTTP ve verzi 1.1, která byla představena v RFC 2817. Tato metoda ale není příliš podporována prohlížeči, takže se pro zabezpečenou komunikaci častěji využívá HTTPS.

Role W3C (World Wide Web Consortium)

Mezinárodní konsorcium, jehož členové společně s veřejností vyvíjejí webové standardy pro World Wide Web. Bylo založeno v roce 1994. Cílem konsorcia je *„Rozvíjet World Wide Web do jeho plného potenciálu vývojem protokolů a směrnic, které zajistí dlouhodobý růst Webu“*. Zabývá se také vzděláním a přístupností, vyvíjí software a nabízí otevřenou diskuzi o Webu prostřednictvím fóra. Úkolem konsorcia je vydávat specifikace (doporučení), kterými by se poté měli řídit tvůrci webových prohlížečů a ostatních programů, které pracují s webem. Před založením konsorcia nabízely různé firmy různé upravené verze jazyka HTML které byly nekompatibilní s verzemi od ostatních výrobců. Konsorcium sjednotilo verze od různých výrobců a dohodlo se s nimi na základních principech a komponentách nových standardů.

Na půdě konsorcia tak vznikla řada technologií - např. kaskádové styly (CSS), XML, formát pro publikaci vektorových obrázků na webu SVG a mnoho dalších.

Vývoj specifikací W3C trvá velice dlouho, protože každý standard (doporučení) prochází několika fázemi (od pracovního návrhu (Working Draft), LastCallWorkingDraft, Candidate Recommendation, Proposed Recommendation až po W3C Recommendation). Doporučení je pak postupně revidováno prostřednictvím samostatně vydávaných errat (*Errata*). Když se nashromáždí větší množství těchto úprav, je vydána nová edice doporučení. Než doporučení projde do závěrečné fáze, může to trvat i několik let. Díky tomu je konsorciu často vyčítána malá pružnost a neschopnost reagovat na aktuální situaci v rychlém vývoji moderního webu.

Popis HTML (tagy párové a nepárové, příklady)

Zdrojový kód HTML je složen z textu a **značek** (tagů). Všechny značky jsou uzavřeny ve špičatých závorkách < a >. Tagy mohou obsahovat parametry - ty nazýváme **atributy** a nachází se mezi špičatými závorkami hned za jménem značky.

```
<jmenoZnacky atribut1="hodnota atributu 1" atribut2="hodnota atributu 2">
```

Atributy upravují a nastavují výsledné zobrazení/chování značky. Např. chceme vložit obrázek, proto použijeme značku `<img>`. Když značku zapíšeme do dokumentu v této podobě, nic se víceméně nestane - prohlížeč totiž neví, jaký obrázek má na místě značky zobrazit (zobrazí bílé ohraničené místo, nebo také nic). Proto do značky doplníme atribut `src`: `<img src="obrazek.jpg">`. Jestliže takový obrázek existuje prohlížeč ho zobrazí v místě, kde se ve zdrojovém kódu nachází značka `<img>`. Pokud ale takový obrázek existovat nebude, opět se nám nic nezobrazí. Proto můžeme tag rozšířit o další atribut `alt`: `<img src="neexistujici-obrazek.jpg" alt="Obrázek stromu">`. Atribut `alt` obsahuje tzv. **alt** ernativní text - tzn. text, který se zobrazí místo obrázku, pokud daný obrázek nebude existovat. Nyní se nám místo bílého místa (v případě neexistující obrázku) zobrazí text "Obrázek stromu". Pomocí atributů lze upravit chování řady značek v HTML kódu.

Tagy se můžou vyskytovat v párech `<něco>` a `</něco>`. Takové tagy nazýváme jako **párové**. Ten první tag něco začíná a druhý něco končí. Lomítko znamená, že jde o **uzavírací tag** (ten bez lomítka je **otevírací**). Příklady párových tagů:

- `<head>` je začátek hlavičky a `</head>` je její konec
- `<body>` je začátek těla stránky a `</body>` je konec těla stránky
- `<b>` je začátek tučného textu a `</b>` je konec tučného textu
- atd.

Tagy, které nemají uzavírací tag, nazýváme jako **nepárové**. Párové tagy něco ohraničují/vyznačují (např. tučný text) - **nepárové** zastupují ve stránce nějaký prvek - např. obrázek, horizontální čáru, apod., nebo mají vztah k celému dokumentu (připojují styly CSS, sdělují doplňující informace o stránce - autor, popis, znaková sada, apod.). Příklady nepárových tagů:

- `<img src="obrazek.jpg">` na dané místo ve stránce vloží obrázek "obrazek.jpg"
- `<hr>` vloží na dané místo horizontální čáru
- `<meta name="description" content="Popis mé skvělé webové stránky">` - obsahuje popis stránky

- `<meta name="author" content="John Doe">` - sděluje informaci o tom, kdo je autorem stránky
- `<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">` - sděluje prohlížeči jakou stránka využívá znakovou sadu (kódování)

Struktura HTML dokumentu

Na začátku každého HTML dokumentu je tzv. **DOCTYPE**. Ten oznamuje prohlížeči, kterou konkrétní verzi HTML používáme. Za ním následuje element `html` a v něm vnořené elementy `head` a `body`.

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN">
<html>
    <head>
        <title>Titulek stránky</title>
    </head>
    <body>
        Vlastní obsah stránky.
    </body>
</html>
```

DOCTYPE obsahuje nejrůznější informace, např. podle jakého DTD (viz XML) je HTML stránka napsána. V případě HTML na DOCTYPE, ale tolik nezáleží, ve většině případů ho prohlížeče ignorují.

> *... Jediné, na čem záleží, je to, aby se stránka vykreslila v IE stejně jako v jiných prohlížečích. Toho se dosáhne tím, že se do stránky přidá libovolná doctype deklarace, která Explorer přepne do standardního módu.*
> Zdroj: http://www.jakpsatweb.cz/doctype.html

Nejnovější HTML5 používá jako DOCTYPE prostý zápis `<!DOCTYPE html>`.

Hlavička stránky (element `head`) obsahuje informace o připojených stylech (CSS), skriptech (JavaScript), informace o autorovi, klíčová slova, stručný popis stránky, titulek stránky, použité kódování (UTF-8, windows-1250, ...), apod. Vlastní obsah stránky, který vidí návštěvník webu je umístěn v elementu `body`.

CSS

Kaskádové styly (v anglickém originále Cascading Style Sheets se zkratkou CSS) je jazyk pro popis způsobu zobrazení stránek napsaných v jazycích HTML, XHTML nebo XML.

Jazyk byl navržen standardizační organizací W3C, autorem prvotního návrhu byl Håkon Wium Lie. Byly vydány zatím dvě úrovně specifikace CSS1 a CSS2, dne 7. června 2011 byla dokončena revize CSS 2.1 a pracuje se na verzi CSS3. Hlavním smyslem je umožnit návrhářům oddělit vzhled dokumentu od jeho struktury a obsahu. Původně to měl umožnit už jazyk HTML, ale v důsledku nedostatečných standardů a konkurenčního boje výrobců prohlížečů se vyvinul jinak. Starší verze HTML obsahují celou řadu elementů, které nepopisují obsah a strukturu dokumentu, ale způsob jeho zobrazení. Z hlediska zpracování dokumentů a vyhledávání informací není takový vývoj žádoucí.

Výhody

- rozsáhlejší možnosti formátování oproti vizuálním elementům v HTML
- jednodušší údržba webové prezentace. Pokud chceme změnit nějaký detail, jako třeba barvu nadpisu, nemusíme složitě procházet HTML kód nebo různé HTML šablony, ale můžeme změnit pouze jednu vlastnost v CSS souboru, který je přilinkován ke všem stránkám.
- **oddělení struktury a stylu** - v jednom (HTML) dokumentu budeme mít pouze sémanticky označen obsah a v druhém (CSS) dokumentu máme definice vzhledu stránek. Tím dosáhneme snadnějšího strojového zpracování samotného obsahu stránek, do kterého se nám nepletou prvky definující vzhled.
- cachování stylů – webový prohlížeč si může soubor se styly uložit do cache paměti, čímž může být dosaženo zrychlení načtení stránky. Na druhou stranu při použití externího CSS souboru dochází k dalšímu HTTP požadavku navíc oproti tomu, když bychom použili buď online CSS nebo přímo formátování HTML
- CSS vlastnosti jednotlivých elementů můžeme dynamicky měnit pomocí Javascriptu
- můžeme určit styl pro různá **výstupní zařízení** - máme tak možnost nadefinovat jiný styl pro tisk, projekt, mobil, PDA či webový prohlížeč. Specifikace CSS nezapomínají ani na zrakově postižené – je možno napsat styly pro hlasový syntetizátor nebo hmatovou čtečku Braillova písma
- koncový uživatel si může napsat svůj vlastní CSS styl pro libovolnou stránku. Většina prohlížečů nějakým způsobem podporuje uživatelské styly, takže uživatel si může například nastavit, aby měl všechny odkazy na všech webech vždy podtržené nebo aby na tomto konkrétním webu mělo písmo vždy černou barvu

Formuláře na webu

Formuláře jsou nezbytnou součástí www stránek. Používají se například pro získání názoru čtenáře stránky, slouží k odesílání vzkazů z www stránek apod. Formulář je obsažen v HTML elementu `<form>` a obsahuje různá vstupní pole (textová políčka, rozbalovací seznamy, zaškrtávátka, přepínače, textová pole, pole pro hesla, pro výběr souboru, apod.) a tlačítka. Vstupní pole slouží k vyplnění a zadání daných údajů do formuláře, tlačítka slouží např. pro odeslání formuláře na server.

Formuláře neodvratně patří k modernímu webu a hlavně k webových aplikacím. Můžou mít různou podobu a účel - např. registrační formulář, formulář pro přihlášení k e-mailu, anketní formulář, formulář pro napsání diskusního příspěvku, apod.

Data z formuláře by měla být vždy nějak zpracována - jinak přítomnost formuláře postrádá smysl. Data z formuláře bývají často odeslána na server, kde jsou zpracována nějakým programem nebo skriptem (např. registračním PHP skriptem). Pro odeslání formulářových dat na server se používají 2 metody - **get** a **post**.

- **metoda GET** - odešle data na server jako součást adresy HTTP požadavku. Tato data jsou v adrese umístěna za otazníkem. Protože je délka adresy omezena, nehodí se tato metoda pro odesílání většího množství dat. Jednotlivé údaje z formuláře jsou odděleny znakem &. Příklad adresy s odeslanými daty - `http://example.com/zbozi.php?hledej=televize&serad=sestup&maxcena=5000`. Data v adrese jsou na straně serveru zpracována nějakým programem nebo skriptem. Adresa s daty je normálně dostupná v historii prohlížeče - to může být nevýhoda, pokud chceme přenášená data skrýt.
- **metoda POST** - odesílá data na server v těle HTTP požadavku a ne v jeho adrese. To umožňuje odesílat objemnější data (např. soubory), protože nejsme omezeni maximální délkou adresy. Také není přenášená data vidět v historii prohlížeče a v logovacích souborech webového serveru, proxy serverů a dalších zařízeních "po cestě". Tuto metodu se doporučuje používat hlavně v případech, kdy odesílaná data nějakým

způsobem mění stav aplikace/serveru - např. když odesíláme diskuzní příspěvek, mažeme článek z webu, přihlašujeme se k webové aplikaci, apod. Takto se liší od metody GET, která se používá hlavně v případech, kdy chceme nějaká data získat (např. vyhledat všechny televize pod 5000 Kč v e-shopu) a nechceme data (např. v databázi) měnit.

Formuláře také mohou být zpracovány JavaScriptem na straně klienta - v takovém případě se data na server nepřenáší.

Klientské skriptování a DOM

Na straně klienta (webového prohlížeče) je možné spouštět programový kód, který je schopen manipulovat se stránkou, upravovat ji, reagovat na akce uživatele, apod. V oblasti webových stránek je nejrozšířenějším klientským skriptovacím jazykem **JavaScript** (v IE jeho obdoba JScript). Je možné použít i jiné jazyky, ty ale nemají takovou podporu v prohlížečích a díky jejich minimálnímu využívání je lze zanedbat (patří sem např. některé jazyky z dílny Microsoftu, které jsou podporovány pouze v Internet Exploreru).

Jelikož se jedná o skriptovací jazyky (a ne jazyky kompilované) je na straně klienta nutná přítomnost **interpretu**, který je schopen do stránky vložený kód načíst a vykonat dané příkazy. Takový interpret bývá zabudován do webového prohlížeče. V posledních letech vedou tvůrci webových prohlížečů zuřivý boj o to, kdo z nich vytvoří rychlejší a méně chybový interpret JavaScriptu a tím nabídnou svých uživatelům možnost používat stále náročnější JavaScriptové aplikace. Díky JavaScriptu lze v dnešních prohlížečích editovat fotografie, psát dokumenty, vytvářet tabulky jako v Excelu, editovat prezentace, malovat, hrát hry (existují např. JavaScriptové emulátory pro staré hry - např. emulátor původního DOOMa). Spousta společností a služeb využívá JavaScript ve svých aplikacích - bez JavaScriptu si neškrtnete např. na Facebooku, Twitteru a spoustě dalších webových stránkách.

Využití

- kontrola formulářů
- nejrůznější efekty na stránkách
- potřeba okamžitě reagovat na podměty uživatele
- manipulace s DOM
- atd.

DOM (Document Object Model - objektový model dokumentu)

DOM je objektově orientovaná reprezentace XML nebo HTML dokumentu. DOM je API (aplikační rozhraní) umožňující přístup či modifikaci obsahu, struktury, nebo stylu dokumentu, či jeho částí. Původně měl každý webový prohlížeč své vlastní specifické (a s ostatní nekompatibilní) rozhraní pro manipulaci s HTML elementy pomocí JavaScriptu. Proto došlo v rámci W3C ke standardizaci tohoto rozhraní a vznikl tak W3C Document Object Model (zkráceně W3C DOM). Tato specifikace je platformně a jazykově nezávislá.

DOM umožňuje přístup k dokumentu jako ke **stromu**, což je zároveň datová struktura používaná ve většině

XML parserů a XSL procesorů.

Specifikace W3CDOM je rozdělena úrovní (levels). Každá úroveň specifikuje určité požadavky a podmínky na programy, které chtějí prohlásit, že podporují daný DOM level. Programy pak musí implementovat všechny požadavky této úrovně a úrovní nižších. Např. pokud chce aplikace prohlásit, že podporuje DOM level 1 musí umět navigaci v DOM dokumentu a manipulaci s jeho obsahem (včetně přidávání elementů).

Pomocí DOM tak můžeme v JavaScriptu vybrat konkrétní elementy a dále s nimi manipulovat, upravovat a mazat je nebo vkládat zcela nové elementy.

ActiveX, Java Applet

Kromě JavaScriptu je možné vložit do stránky také jiné objekty, které jsou schopné se stránkou nějakým způsobem manipulovat. Na rozdíl od JavaScriptu se nejedná o skriptovací jazyky, ale o externí programy, většinou zkompilované do podoby nějakého balíčku, který je vložen do webové stránky. Mezi tyto technologie patří např. **ActiveX** od Microsoftu a **Java applety**.

ActiveX

ActiveX je technologie, kterou vyvinula společnost Microsoft pro sdílení informací mezi různými aplikacemi. ActiveX spolupracují pouze s aplikacemi, jako je Microsoft Word, Excel, Internet Explorer, PowerPoint a pracují pouze na počítači s operačním systémem Windows.

Nejznámější použití ActiveX je tzv. ActiveX Controls – jde o malé soubory kódů, které si mohou uživatelé prohlížeče Internet Explorer stáhnout a spustit v počítači. ActiveX Controls jsou psané v běžných programovacích jazycích, jako je Visual Basic a C++. Nejedná se o samostatné aplikace a mohou být spuštěny pouze z hostitelské aplikace jako například Internet Explorer, MS Office apod. ActiveX Controls lze přirovnat k Java Appletům, na rozdíl od nich však mají plný přístup k operačnímu systému, neboť jde o objekty COM a mají tak neomezený přístup k počítači. ActiveX Controls mohou přistupovat k místnímu systému souborů a měnit nastavení registru operačního systému.

Na webových stránkách lze ActiveX komponenty používat od roku 1996, kdy byla tato představena a zároveň byla vydána nová verze Internet Exploreru - verze 3.0. ActiveX komponenty jsou objekty postavené na technologii COM. Prakticky jakákoliv aplikace může být vytvořena v podobě ActiveX componenty. Lze implementovat jednoduché objekty zobrazující prostý text i složité objekty typu tabulek programu Excel apod.

Na stránku můžeme ActiveX objekt umístit pomocí tagu `<object>`, jemuž nastavíme parametr `classid`. Ten obsahuje hodnotu GUID (Global Unique IDentifier) – jednoznačné hexadecimální číslo identifikující COM objekt. Seznam objektů se na klientovi uchovává v systémovém registru Windows. Pokud na cílovém počítači ještě není přítomen objekt, na který se stránka odkazuje, je prohlížečem stáhnut z adresy dané parametrem CodeBase tagu `<object>` a nainstalován, tzn. jsou přidány záznamy do registru systému. Pak již lze COM objekt spustit a zobrazovat jeho grafický výstup v prohlížeči.

ActiveX jsou často zneužívány k virovým útokům. ActiveX se po stažení stávají součástí operačního systému se schopností manipulace s hardware i software počítače. Microsoft po kritice namísto omezení funkcí ActiveX zvolil v následujících verzích IE jinou cestu zabezpečení. Microsoft vyvinul systém registrací a certifikátů, díky nimž může prohlížeč prvky ActiveX ověřovat ještě před samotným zavedením. Je ponecháno na rozhodnutí

uživatele, zda je ActiveX legitimní, nebo se jedná např. o trojského koně.

ActiveX je dnes již překonaná technologie, která navíc dokáže být nebezpečná a její používání se nedoporučuje.

Podobné funkcionality bývá v ostatních prohlížečích a na ostatních platformách dosaženo pomocí tzv. plug-inů. Je to knihovna, kterou prohlížeč načte při svém spuštění a která modifikuje vnitřní chování prohlížeče. Tímto způsobem přidává prohlížeči novou funkcionalitu a to různého typu. Některé plug-iny lze asociovat s MIME typem. Je-li pak pomocí tagu <embed> vložen do HTML stránky soubor určitého MIME typu, je spuštěn asociovaný plug-in (např. pro přehrávání videí a spuštění flashe (např. pro flash hry) bývá používán plug-in Flash Player, pro zobrazení PDF souborů plug-in Adobe Readeru, apod.).

Java Applet

V programovacím jazyku Java je zdrojový kód nejprve napsán jako jednoduchý text a uložen do souborů s koncovkou `.java`. Tyto zdrojové soubory se potom kompilujjí do souborů s koncovkou `.class` pomocí Java kompilátoru. Soubor s koncovkou `.class` však neobsahuje kód, který dokáže přímo zpracovávat procesor; místo toho obsahuje tzv. bajtkód (bytecode) - instrukce pro virtuální počítač Java Virtual Machine (Java VM, JVM). Při spuštění Java programu, tak dojde ke spuštění JVM - pomocí něho je pak program spouštěn (interpretován). Java tak kombinuje vlastnosti kompilovaných jazyků (kompilace - program je rychlejší, než kdyby byl interpretován) a jazyků skriptovacích (interpretovaných) - snadná přenositelnost a nezávislost na kontrétní platformě či operačním systému.

Applety jsou malé programy, jejichž soubor `.class` s byte kódem je uložen na serveru - vždy je ale vykonáván na straně klienta v rámci jiného programu (nejčastěji webového prohlížeče). Aby bylo možno používat applety, musí být na klientském počítači instalován Java Virtual Machine a prohlížeč je musí podporovat.

Abychom v prohlížeči mohli spustit applet, vložíme do stránky HTML tag `<applet>` s atributem, který specifikuje umístění appletu na serveru. Jakmile prohlížeč načte tento tag, spustí tzv. Class Loader (součást JVM), jenž ze serveru stáhne kód appletu a kód všech potřebných tříd, na něž se applet odkazuje a které ještě nemá klient v počítači. Jsou-li všechny požadované soubory dostupné, je applet spuštěn.

Aby byla zajištěna nezbytná bezpečnost při běhu appletu, omezuje dostupnou funkcionalitu tzv. Security Manager, který umožňuje uživateli zvolit, jaké skupiny funkcí mají být povoleny či zakázány. Obvykle nemají applety přístup do souborového systému a nemohou navazovat spojení s jinými servery, než odkud byly stáhnuty.

Applety se používají především k implementaci složitějšího grafického prostředí na straně klienta, než je schopna poskytnout technologie dynamického HTML. Za tím účelem je appletu na stránce HTML vymezen obdélníkový prostor, jehož rozměry jsou dány dalšími atributy tagu `<applet>`. V této oblasti je veškeré vykreslování ponecháno na appletu a prohlížeč ho neovlivňuje.