

Tvorba webových stránek

Ing. Radek Burget, Ph.D.
burgetr@fit.vutbr.cz

Osnova

1. **5.2.** Internet a služba WWW
2. **12.2.** Úvod do HTML
3. **19.2.** Úvod do kaskádových stylů (CSS)
4. **26.2.** Kaskádové styly – základní mechanismy
5. **4.3.** Kaskádové styly – rozmísťování prvků
6. **11.3.** Praktické použití, layouty

Osnova

8. **18.3.** *Půlsemestrální test*
9. **25.3.** Pokročilé vlastnosti HTML 5, XHTML
10. **1.4.** Přístupnost webu
11. **8.4.** JavaScript a jQuery
12. **15.4.** JavaScript a jQuery
13. **22.4.** Rozšiřující technologie, CSS frameworky
14. **29.4.** *Zápočtový test*

Cvičení

- Praktické vyzkoušení webových technologií
 - Celkem 6 týdnů
- Dva samostatné projekty
 1. Základní prezentace v HTML + CSS
 2. Pokročilá prezentace s využitím HTML5, CSS3 a jQuery
- Hodnocení
 - Cvičení – **10 bodů**
 - Projekty – **20 + 30 bodů**
 - Testy (v rámci přednášek) – **20 + 20 bodů**

Plán cvičení

Cvičení v týdnu od

1. **10.2.** DNS, HTTP
2. **17.2.** Základy HTML
3. **24.2.** Základní styl pomocí CSS
4. **2.3.** Rozložení stránky pomocí CSS
5. **6.4.** CSS3
6. **20.4.** JavaScript & JQuery

Osoby a obsazení

- Přednášky
 - Radek Burget, burgetr@fit.vutbr.cz
<https://www.fit.vut.cz/person/burgetr/>
 - Jiří Hynek, ihynek@fit.vutbr.cz
<https://www.fit.vut.cz/person/ihynek/>
- Cvičení
 - viz stránka předmětu

Tvorba webových stránek

Tvorba webových stránek

- Webové prezentace: výkladní skříň firmy (organizace, ...)
- Webdesign: návrh a realizace webové prezentace
- Zahrnuje větší množství úkolů:
 - Vstupní analýza
 - Návrh informační architektury
 - Grafický návrh
 - Tvorba obsahu
 - Realizace (kódování) návrhu
 - Testování
 - Nasazení na web
 - Monitorování, budování návštěvnosti
- Obvykle práce většího týmu odborníků

Profese webdesignu

- Konzultant
 - Co vlastně zákazník chce a co potřebuje?
- Copywriter
 - Jak napsat text tak, aby prodával?
- UX (user-experience) designer
 - Jak zákazník najde to, co potřebuje(me)?
- Grafik
 - Jak obsah prodat vizuálně?

Profese webdesignu (II)

- Kodér
 - Jak všechno výše uvedené dostat do prohlížeče uživatele?
- SEO konzultant, linkbuilder
 - Jak přitáhnout návštěvníky?
- Marketingový konzultant
 - Jak na tom vydělat?

Principy WWW

WWW

- WWW = ***World Wide Web***
- Hlavní rysy webu
 - **Distribuovaný**
 - Vytvořen pomocí velkého množství nezávislých jednotek
 - **Heterogenní**
 - Na různých platformách
 - **Dynamický**
 - Neustále a nekontrolovatelně se mění
 - **Orientovaný na dokumenty**
 - Základní datovou jednotkou je **dokument**

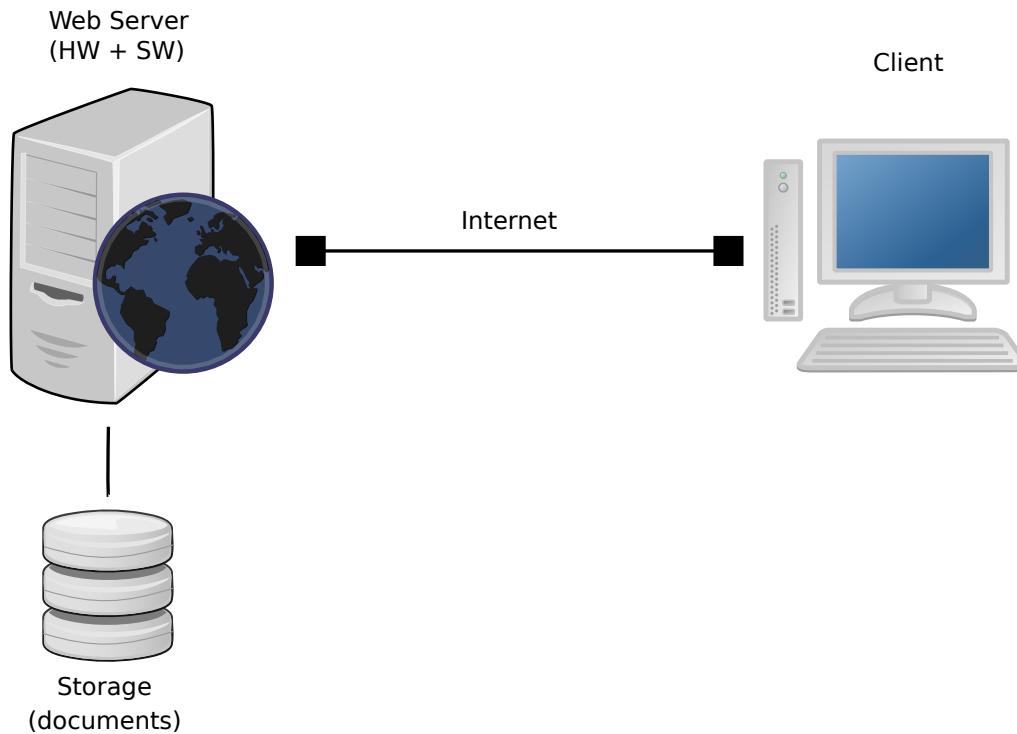
Historie

- **1989** - Tim Berners-Lee (CERN) publikuje článek „***Information Management: A Proposal***“ - návrh architektury a využití hypertextu
- **1990** - Start projektu „World Wide Web“, první stejnojmenný prohlížeč a první webová stránka
- **1992** - 26 rozumně fungujících serverů
- **1993** - Přes 200 serverů (převážně akademických), první alfa verze prohlížeče **Mosaic**
- **1994** - Založení konsorcia W3. Zatížení prvního serveru **info.cern.ch** 1000x větší než na počátku.

Historie

- **1996** - oficiální specifikace HTML 3.2 (rámy, skripty, vkládání objektů)
- **2000** - první specifikace XHTML 1.0 (verze HTML založená na XML)
- **2004** - vznik pracovní skupiny výrobců prohlížečů WHATWG
- **2007** - spojení W3C a WHATWG do WG, počátky práce na HTML5
- **2014** - dokončení specifikace HTML 5

Architektura WWW



WWW server

- Úložiště dokumentů
 - Organizované hierarchicky v adresářové struktuře
 - Např. **/products/phones/nokia.html**
- Software běžící na fyzickém serveru
 - Na požadání je schopen zaslat libovolný dokument
 - Např. Apache, Microsoft IIS (Internet Information Services) server, ...

Ostatní služby serverů

- Na jednom serveru mohou běžet další služby
- Každá služba je identifikována číslem (**port**) a jménem
- Příklady:

Port	Jméno	Protokol
21	ftp	FTP
22	ssh	SSH
23	telnet	-
25	smtp	SMTP
80	www	HTTP
443	https	HTTPS

WWW klient – prohlížeč

- Vysílá požadavek na server a zobrazí získaný dokument.
- Zobrazovací jádra:
 - Gecko (Mozilla Foundation)
 - Firefox
 - Trident (Microsoft)
 - Internet Explorer
 - WebKit (Open source, KHTML + Apple)
 - Safari, dříve Chrome
 - Blink (Google)
 - Chrome, Opera

Podrobný přehled

Dokumenty na WWW

- Základní jednotka informace na WWW = **dokument**
 - Soubor uložený na serveru
 - Jednoznačně identifikován pomocí **URI**
- Různé typy dokumentů
 - Textové dokumenty
 - **Hypertextové dokumenty**
 - Obrázky
 - Multimediální data (zvuk, hudba, video, ...)
 - Programy
 - ...

Identifikace dokumentu – URI

- URI = ***Unified Resource Identifier***
- Jednoznačně identifikuje jeden dokument na WWW
- Typický tvar

http://www.fit.vutbr.cz/units/UIFS/index.php
Schéma _____
Jméno hostitele _____ Cesta k dokumentu
_____ HTTP adresa

Identifikace dokumentu – URI

- Port je možno zadat za jménem hostitele

```
http://www.fit.vutbr.cz:8080/document.html
```

- Jméno souboru nemusí být zadáno

```
http://www.fit.vutbr.cz/  
http://www.fit.vutbr.cz/study/
```

- URL = ***Unified Resource Locator*** – neoficiální, ale běžně užívaný termín
- V technických specifikacích se ve stejném významu používá URI

Hypertext

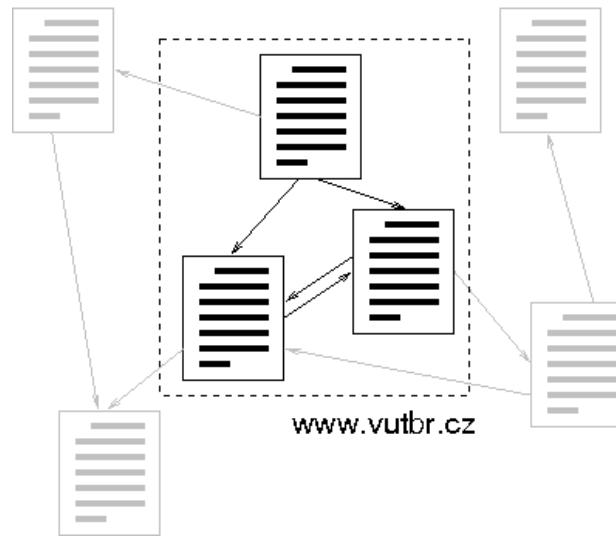
- ***Text neomezený linearitou***
- Umožňuje vkládat do textu odkazy na jiné dokumenty
- **Odkaz** je souvislá část textu s přiřazenou cílovou adresou
- Při **aktivaci odkazu** se zobrazí cílový dokument
- Hovoříme o **textu odkazu** a **adrese odkazu**
- Pro hypertext kombinovaný s jinými typy dat se často používá název **Hypermedia**

Příklad

Po kliknutí na text odkazu se dostaneme na následující slajd.

Webové místo (website)

- Množina dokumentů, které jsou obvykle na stejném hostiteli a propojeny odkazy tak, že tvoří logický celek
- Např. www.vutbr.cz



Komunikace klient-server

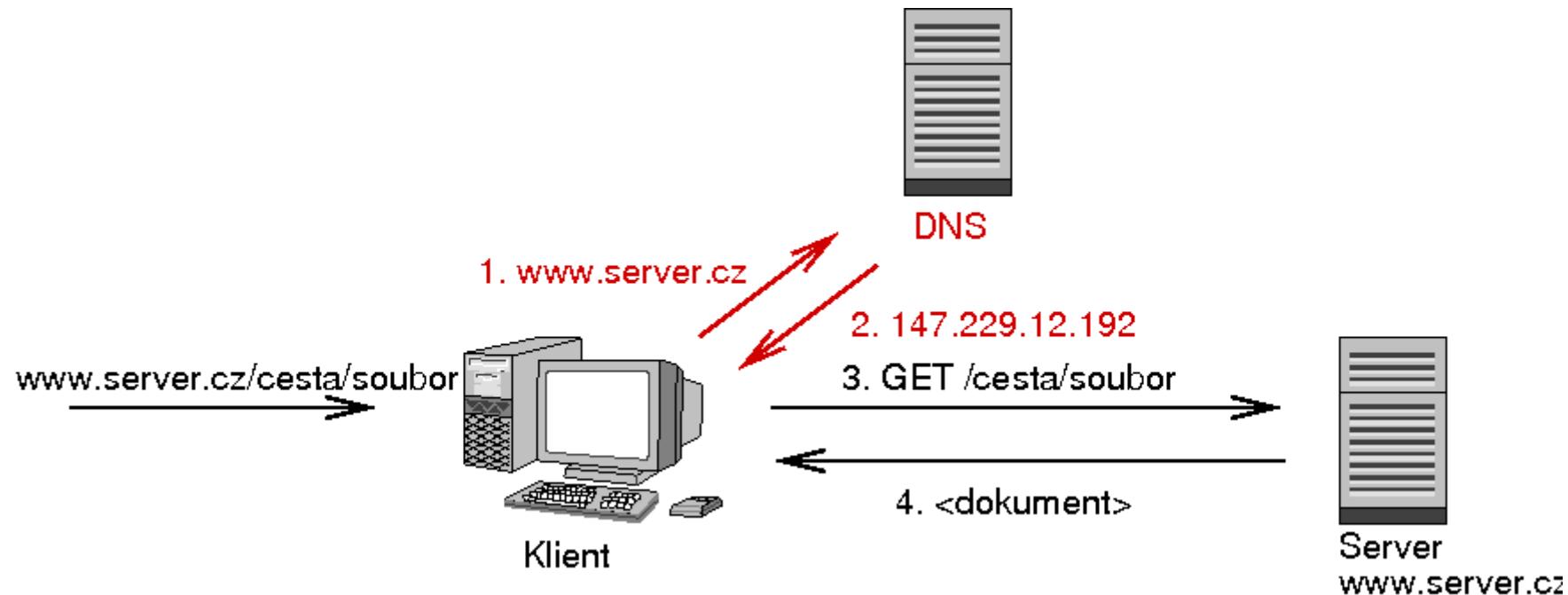
Způsob přenosu dokumentu je podrobně definován v mnoha vrstvách:

- Fyzická + linková – libovolná
 - ethernet, ATM, wifi, ...
- Sítová + transportní – **TCP/IP**
 - Zajišťuje bezchybný přenos dat mezi dvěma body
 - Definuje jednotný tvar unikátních adres počítačů (IP adresy)
- Aplikační – převážně **HTTP**
 - HyperText Transfer Protocol
 - Definuje tvar požadavku na dokument
 - Definuje tvar odpovědi (požadovaný dokument nebo chyba)
 - Kódy chyb

Protokol HTTP

- Založen na principu dotaz - odpověď
- Server neuchovává informace o předchozích dotazech
 - => protokol je **bezstavový**
- Historie
 - **Verze 0.9** – pouze přenos dokumentů
 - **Verze 1.0** – MIME identifikace typu dokumentu
 - **Verze 1.1** – Trvalé spojení, vyjednávání o obsahu, další rozšíření
 - **HTTP/2** – Komprese hlaviček, řetězení požadavků, HTTP/2 push, efektivita (asi 10% webů v prosinci 2016)

Dotaz HTTP



Metody protokolu HTTP

- „Příkaz“ zasílaný serveru
- Určuje druh požadované služby
- Server nemusí podporovat všechny metody

Metoda	Popis
GET	Požadavek na dokument (URL)
HEAD	Jako GET, pouze hlavička odpovědi
POST	Zpracování dat v požadavku
PUT	Uložení dokumentu na server

HTTP požadavek

- Dotazový řádek
- Hlavíčka
- Prázdný řádek
- (Tělo dotazu)

```
GET /index.html HTTP/1.0
Accept: text/html
Accept: image/gif
User-Agent: Mozilla/4.0 (compatible; MSIE 6.0; Windows NT
5.1)
```

Druhy odpovědí

- Různé odpovědi jsou označeny jménem a číselným kódem
- **1xx** - informační (zatím nevyužito)
- **2xx** - úspěch
 - **200** OK
 - ...
- **3xx** - je třeba další akce (přesměrování)
 - **301** Moved permanently
 - **302** Moved temporarily
 - ...

Druhy odpovědí

- **4xx** - chybný požadavek
 - **400** Bad request (server nerozumí)
 - **401** Unauthorized (uživatel není oprávněn)
 - **403** Forbidden (server není oprávněn)
 - **404** Not found
 - **406** Not acceptable (požadovaná varianta není k dispozici)
 - ...
- **5xx** - chyba na straně serveru
 - **500** Internal server error
 - **503** Service unavailable (přetížení, ...)
 - **505** HTTP version not supported
 - ...

Odpověď'

- Stavový řádek
- Hlavička
- Obsah odpovědi (oddělený prázdným řádkem)

```
HTTP/1.1 200 OK
```

```
Date: Wed, 08 Sep 2004 13:19:30 GMT
```

```
Server: Apache/1.3.31 Ben-SSL/1.55 (Unix)
```

```
Pragma: no-cache
```

```
Connection: close
```

```
Content-Type: text/html; charset=iso-8859-2
```

```
Content-Language: cs
```

```
<html>
```

```
....
```

Zpracování dokumentu klientem

- **Dokumenty HTML a XML**
 - Interpretuje a zobrazí (rendering)
 - Zobrazení může vést na další HTTP dotazy (obrázky, ...)
- **Textové soubory**
 - Přímo zobrazí
- **Bitmapové obrázky (JPG, PNG, GIF)**
 - Přímo zobrazí
- **Ostatní**
 - Předá pomocnému programu
 - Může mít formu **zásvněho modulu** (plugin)
- **Pokud všechno selže**
 - Alespoň uloží na disk apod.

MIME typ

- Typ dokumentu je určen pomocí standardu MIME
 - Specifikace ve tvaru **třída/typ**
 - Např. text/plain, text/html, image/jpeg, video/mpeg, ...
- Informaci o typu obvykle zasílá server spolu s dokumentem
 - Hlavice **Content-Type**:
 - Její určení závisí na serveru a jeho nastavení
- Pokud ji nezašle, klient se pokusí určit typ sám (např. podle přípony)

Požadavky na klienta

- Klient musí umět zpracovat (zobrazit) získané dokumenty
- Z pohledu tvůrce dokumentů: dokumenty musí být takové, aby je **předem neznámý** prohlížeč dokázal zobrazit
- Problém nových technologií
 - Při tvorbě obsahu je nutno pamatovat na různé prohlížeče
 - Technologie je použitelná, pokud ji podporuje většina prohlížečů
- Prohlížeče nejsou zcela kompatibilní
 - Chyby a nedostatky jsou většinou již známé a popsané
 - Je třeba s nimi počítat
 - Vina za chybné zobrazení vždy padá na tvůrce stránek

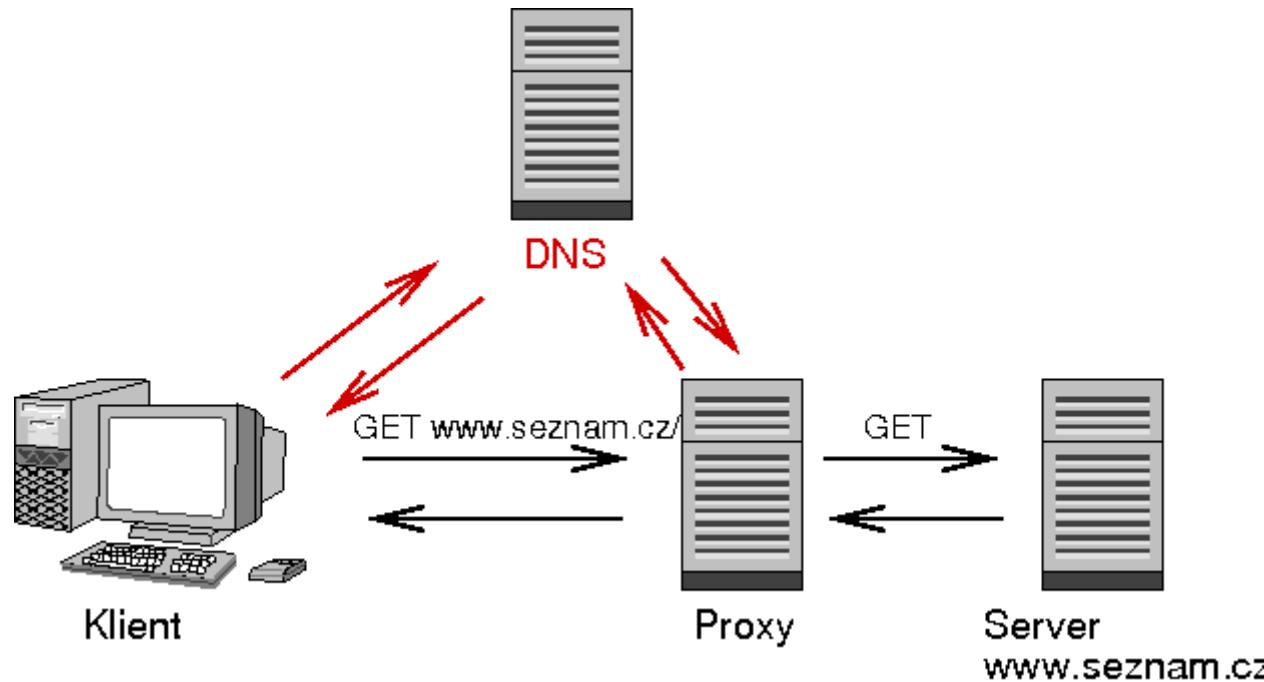
Další adresní schémata

- **http:** – použije se protokol HTTP (služba www)
`http://www.fit.vutbr.cz/news`
- **https:** – zabezpečený HTTP
- **ftp:** – file transfer protocol
`ftp://ftp.fit.vutbr.cz/pub/data/texty.zip`
- **mailto:** – adresa elektronické pošty
`mailto:burgetr@fit.vutbr.cz`
- **file:** – lokální souborový systém
`file:///home/burgetr/text.html`
`file:///C:\My Documents\text.html`

Cache

- Bud' samostatný počítač nebo v rámci klienta
 - Samostatný program
 - Součást prohlížeče
- Jednou získané dokumenty (hypertext, obrázky, ...) se ukládají
- Při novém požadavku se ověří, zda došlo ke změně
 - Požadavkem HEAD
 - Eviduje se doba platnosti
- Znovu se přenáší jen změněné dokumenty
- Doba platnosti a způsob chování cache lze nastavit na serveru nebo v každém dokumentu

Proxy server



- Nastaveno v klientském prohlížeči
- Možno specifikovat i jednotlivě

Použití proxy

- Zprostředkování přístupu
 - Nelze zajistit přímé připojení klienta k Internetu
- Omezení přístupu
 - Omezení připojení pouze na WWW
 - Filtrování dostupných stránek
 - Evidence přístupů
- Vyšší efektivita
 - Vyrovnávací paměť – **Cache**

Cache server

- Na FIT cache.fit.vutbr.cz
- Speciální program (např. Squid)
- Funguje jako proxy server
- Vyhodnocuje GET požadavky
 - Rozhoduje, zda dokument stáhne znovu nebo vrátí odpověď z disku
- Efektivnější než cache v prohlížeči (větší cache, sdílení)
- Transparentní cache
 - Zdánlivě bez proxy
 - HTTP provoz se "odchytává" a přesměrovává na routeru

Ovládání cache

- Některé dokumenty se nemění, není nutné je pořád stahovat znovu
 - Manuály, obrázky, ikony, ...
- Jiné se mění neustále - je třeba vždy stáhnout aktuální verzi
 - Stránky s aktualitami, ...
- Pro dokumenty lze nastavit
 - Dobu platnosti (do kdy se má uchovávat v cache)
 - Zapnout / vypnout ukládání
- Nastavení lze provést
 - Nastavením HTTP serveru
 - V některých dokumentech přímo jako součást obsahu

Dynamické stránky

- Statické stránky
 - Obsah je předem připraven a uložen na server
 - Při prohlížení se přenesou ze serveru a zobrazí
- Dynamické stránky
 - Dokument (nebo jeho část) je výsledkem provedení programového kódu
 - Kód je uložen na serveru a je proveden
 - Na serveru při obdržení požadavku na dokument
 - U klienta (v prohlížeči) při obdržení dokumentu, který obsahuje vložený kód
- Webová aplikace
 - Funkčně propojená množina dynamických stránek

Stránky generované na serveru

- Dokument je generován v okamžiku obdržení požadavku
- Možnost dodání vstupních parametrů (HTTP metoda GET nebo POST)
- Výhody
 - Není nutná podpora na straně klienta
 - Veškerá technologie na straně serveru (databáze, ...)
- Nevýhody
 - Větší zatížení serveru
 - Při každé změně parametrů se přenáší celá stránka
- Příklady
 - CGI, PHP, ASP(.NET), JSP, ...

Stránky generované v prohlížeči

- Stránky obsahují podprogramy v některém jazyce (JavaScript)
- Prohlížeč během zobrazování stránky vykoná vložený kód
- Možno reagovat na vnější událost (pohyb myši, ...)
- Výhody
 - Rychlosť - stránku lze změnit bez přenášení dat
 - Možnost interaktivní práce
- Neýhody
 - Klient musí umět kód interpretovat
 - Problémy s kompatibilitou
 - Bezpečnostní problémy

Systémy pro správu obsahu

- Content Management Systems (CMS)
- Dynamicky generují stránky na základě specifikace:
 - Obsah stránek
 - Šablony vzhledu
 - Propojení dokumentů (menu, odkazy)
- Umožňují svěřit správu obsahu třetí osobě
- Vyšší nároky na implementaci stránek a provoz

Odkazy

- HTTP
 - Oficiální specifikace ([1.0](#), [1.1](#), [2](#))
- Prohlížeče
 - [Mozilla \(+Firefox\)](#)
 - [Opera](#)
 - [Chrome](#)
- Cache
 - [Squid](#)

2. Značkovací jazyky: HTML

Značkovací vs. klasické jazyky

- **Klasické programovací jazyky** - převládá strukturovaný text (programový kód), obecný text se vyskytuje ve formě **řetězců**

```
if (a < 5)
    printf("Zadal jste málo, zadejte více");
```

- **Značkovací jazyky** - převládá obecný text, do kterého vloženy strukturované úseky pomocí **značek**.

Do textu je vložen [odkaz](index.html).

Různé značkovací jazyky

- Liší se syntaxí a způsobem využití
- Počítačová typografie
 - TeX (LaTeX) – tvorba publikací
 - Rich Text Format (RTF) – kancelářské dokumenty
 - Markdown – intuitivní značkování pomocí symbolů
- Elektronické manuály
 - nroff, troff, ...
- Webové stránky
 - Rodina XML (XHTML, obecné využití)
 - **Jazyk HTML**

Jazyk HTML

HTML

- HTML = **Hypertext Markup Language**
- Standardní jazyk pro tvorbu hypertextových dokumentů
- Standard spravován konsorciem W3C
- V současné době verze 5 (od 2014)
- Každá webová stránka je primárně tvořena dokumentem v jazyce HTML případně rozšířeným o další data a rozšíření.

HTML dokument

- Obvykle označen příponou **.html** nebo **.htm**
- MIME type **text/html**
- Prostý textový soubor (Unicode)
- Může obsahovat
 - Text
 - Vložené značky
 - Znakové entity

Text dokumentu

- Neformátovaný text
- Kódování znaků: **UTF-8 (ISO-8859-2, windows-1250, ...)**
- Zásady zpracování mezer:
 - Větší počet mezer se redukuje na jednu
 - Konec řádku se převádí na mezeru
 - Prázdné řádky se ignorují
- Řádky se zalamují dynamicky až při vykreslování

Příklad

Skákal pes přes oves
přes zelenou louku.

Šel za ním myslivec...

Se zobrazí jako

Skákal pes přes oves přes zelenou louku. Šel za ním myslivec...

Značky a elementy

- HTML element: úsek dokumentu vymezený **značkami**

```
<p>Obsah elementu</p>

<div class="menu" id="mainmenu">
  Obsah elementu<br> Další obsah elementu.
</div>

<div>Nějaký <em>zvýrazněný</em> text.</div>
```

- Element má jméno, atributy a obsah
 - Z definice prázdné (void) elementy – jen počáteční značka
- Na velikosti písmen nezáleží

Vlastnosti elementů

- Specifikace HTML definuje existující jména elementů.
- Pro každý element je dále definován
 - Jeho význam (sémantika)
 - Kontext, kde může být použit
 - Povolený obsah
 - Povolené atributy
 - ... další detailly
- Příklad: definice elementu
- Automatická kontrola syntaktické správnosti: **validace**
 - Online validátor W3C

Poznámky

```
<!-- Libovolný text -->
<!--
Libovolný text.
Tento text se přeskočí.
-->
```

- Poznámky nemají žádný význam, ale v dokumentu zůstávají
- Vhodné pro „ukrytí“ částí dokumentu, které nemají být zpracovány prohlížečem
 - Poznámky autora stránky
 - Vložené příkazy pro server

Znakové entity

- Pro vkládání speciálních znaků

<	<
>	>
&	&
®	®
©	©
€	€
 	nedělitelná mezera

Ve skutečnosti je jich mnohem více.

Struktura dokumentu

```
<!DOCTYPE html>
<html lang="cs">
  <head>

    ... hlavička ...

  </head>
  <body>

    ... tělo dokumentu ...

  </body>
</html>
```

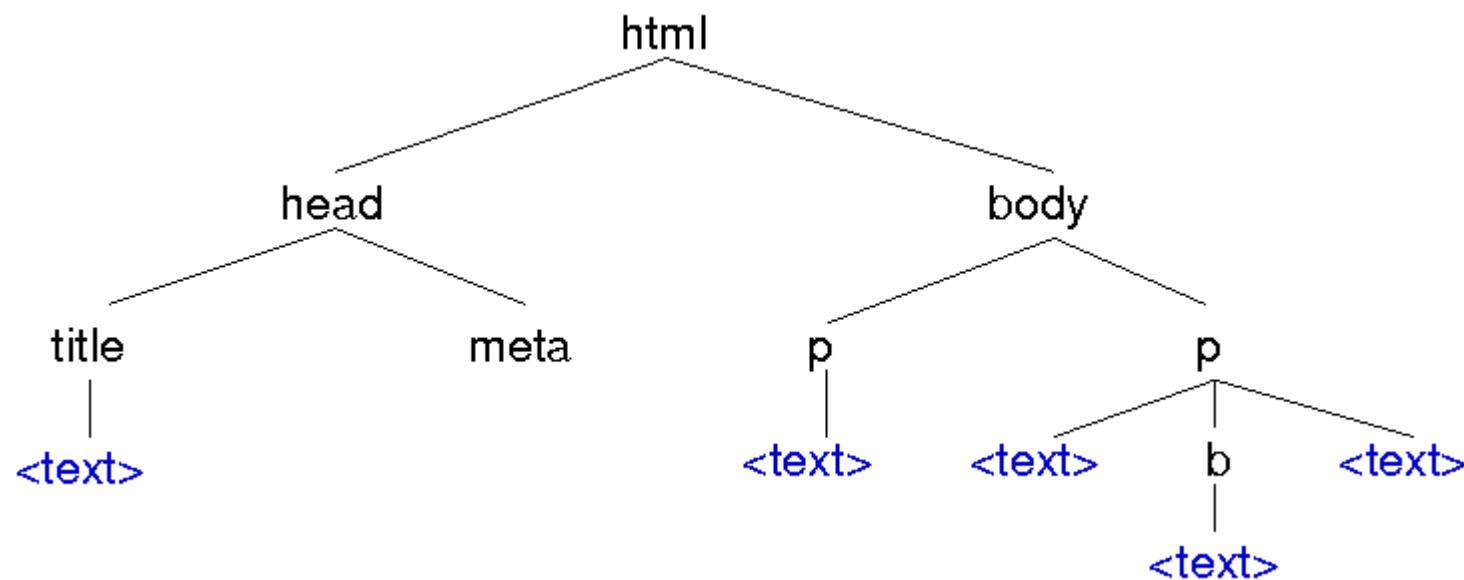
Hlavíčka dokumentu

```
<!DOCTYPE html>
<html lang="cs">
<head>
    <meta charset="utf-8">
        <title>Titulek stránky</title>
</head>
...

```

HTML dokument jako strom

- HTML dokument lze popsat jako **strom elementů**
- Kořenový element je tvořen elementem `<html>`
- Synovské uzly jsou tvořeny elementy bezprostředně vnořenými v otcovském elementu



Obsah dokumentu

- Tvořen elementem **<body>**
- Definuje **obsah** a jeho **strukturu**
- Obsah
 - Textový obsah
 - Obrázky, multimedia, interaktivní prvky, ...
- Struktura
 - Sekce a podsekce
 - Nadpisy více úrovní
 - Odrážkované a číslované seznamy
 - Tabulky
 - Formuláře, ...

Vizuální prezentace obsahu

- Jazyk HTML není určen pro definici vzhledu jednotlivých částí obsahu
 - např. barvy, písma, rámečky, apod.
- Starší verze HTML tuto možnost měly (různé značky a atributy)
 - V HTML 5 kompletně odstraněno
- Vizuální styl je nyní definován oddeleně pomocí Cascading Style Sheets (CSS)

Strukturování obsahu

- Používáme různé elementy definované ve specifikaci HTML
 - Obecně tzv. ***flow elements*** – obsahové elementy
- Každý element má definovaný význam (sémantiku)
 - Např. <p>0dstavec textu</p>
- Řádkové elementy (inline elements, phrasing content)
 - Neovlivňují tok textu (zalomení)
- Blokové elementy (block-level elements)
 - Způsobují zalomení řádku
 - Mohou způsobit odsazení, vynechání místa apod
 - V HTML 5 specifikaci rozděleny na více kategorií

Příklad: řádkový element

- Element **je řádkový**

Některá **důležitá** slova jsou označena.

- zobrazí se:

Některá **důležitá** slova jsou označena.

Příklad: blokový element

- Element `<h1>` je blokový

Předchozí text `<h1>Nadpis</h1>` následující text.

- zobrazí se:

Předchozí text

Nadpis

následující text.

Řádkové elementy

a) Významové odlišení částí textu

< em >	<i>zvýraznění – důraz</i>
< strong >	zvýraznění – důležitost
< b >	zvýraznění – obecné
< i >	<i>odlišený text (styl, jazyk, apod.)</i>
< u >	vizuální <u>označení části textu</u>
< small >	méně významná část textu (pod čarou)
< s >	část textu, která již neplatí
< abbr >	zkratka nebo akronym, např. OSN
< span >	bez významu, možné atributy
< br >	zalomení řádku (prázdný)

Řádkové elementy (II)

b) vzorce, programy apod.

<var>	<i>proměná v programu, vzorci</i>
<sup>	<i>horní^{index}</i>
<sub>	<i>dolní_{index}</i>
<code>	výpis kódu
<dfn>	<i>nový termín</i>
<cite>	<i>citace, odkaz na zdroj (název, jméno, apod.)</i>
<q>	<i>"ocitovaný obsah" (doslovně)</i>

Vnoření značek

```
Normální text <b>Tučný text <i>tučná kurzíva</i>  
jen tučný</b> zase normální.  
<i>Jen kurzíva</i>
```

se zobrazí jako

```
Normální text Tučný text tučná kurzíva jen tučný zase normální.  
Jen kurzíva
```

Odkazy

- Element <a>

Následuje

< a href="http://www.seznam.cz/">odkaz.

Následuje odkaz.

Relativita odkazů

- Na jiný server
 - `href="http://www.vutbr.cz/"`
 - `href="http://www.vutbr.cz/doc/info.html"`
- Na stejný server
 - Absolutní `href="/doc/info.html"`
 - Relativní k umístění souč. dokumentu
 - `href="next.html"`
 - `href="obsah/clanek.html"`
 - `href="..../index.html"`

Kotvy (anchors)

- V dokumentu lze definovat pojmenované místo pomocí atributu **id** libovolného elementu
 - `<p id="jmeno">`
- Na toto místo se odkážeme přidáním **#jmeno** za URL dokumentu
 - `` - současný dokument
 - `` - jiný dokument
 - `` - jiný server

Blokové elementy

- Základní struktura dokumentu
 - Záhlaví, zápatí, sekce, navigace, nadpisy
- Skupiny obsahu
 - Odstavce, seznamy, tabulky, obrázky

Základní struktura obsahu

- **article** – článek, příspěvek (samostatně použitelná část obsahu)
- **section** – kapitola článku, příspěvku, apod.
- **aside** – část, která se jen vzdáleně vztahuje k hlavnímu obsahu
- **header** – záhlaví dokumentu, článku, apod.
- **footer** – zápatí
- **nav** – část obsahující navigaci
- **h1 - h6** – nadpisy
- **div** – obecný blok bez dalšího významu (pro skriptování, styly, apod.)

Strukturování elementů

- Řádkové elementy lze libovolně vnořovat
 - $y = e^x$
 - $y = e^{2x}$
- Řádkový element ***nesmí*** obsahovat blokové
 - V ***tučném*** textu ***< p >*** odstavec ***< /p >*** nemůže být ***< b >***.
- Blokový element může obsahovat všechny řádkové
- Různé blokové elementy mohou obsahovat některé jiné blokové elementy
 - Určeno specifikací
 - Přehledně uvedeno v příloze

Nadpisy

- Až 6 úrovní nadpisů
- Tagy `<h1>` - `<h6>` používat **pouze** pro nadpisy!
- Možné různé kombinace s `<header>` a/nebo `<section>`
- Podtitul apod. se neoznačuje jako nadpis

```
<header>
  <h1>HTML 5.1 Nightly</h1>
  <p>A vocabulary and associated APIs for HTML and XHTML</p>
  <p>Editor's Draft 9 May 2013</p>
</header>
```

Příklad struktury dokumentu

```
<article>
  <header>
    <h1>Titulek</h1>
    <p>Informace...</p>
  </header>
  <section>
    <h2>Sekce 1</h2>
    <p>Lorem ipsum dolor sit amet, adipiscing elit. Quisque a mi.
  </p>
  </section>
  <section>
    ...
  </section>
</article>
```

Příklad navigace

```
<nav>
    <h2>You are here:</h2>
    <ul id="navlist">
        <li><a href="/">Main</a> →</li>
        <li><a href="/products/">Products</a> →</li>
        <li>
            <a href="/products/dishwashers/">Dishwashers</a> →
            <li><a>Second hand</a></li>
        </ul>
    </nav>
```

Odstavce

- Odstavec začíná značkou **< p >**
- Ukončující značka **</ p >** není povinná
 - Pokud chybí, je odstavec ukončen začátkem dalšího blokového elementu
 - Doporučuje se odstavce ukončovat
- Uvnitř odstavce již mohou být pouze řádkové elementy
- Obvykle se zobrazí vertikálně odsazený od okolního textu

Předformátovaný text

- Zachovává formátování pomocí mezer a konce řádku
- Značky se interpretují normálně
- Obvykle neproporcionalním písmem

```
<pre>  
První řádek  
Druhý řádek  
</pre>
```

První řádek
Druhý řádek

Číslované seznamy

1. První položka
2. Druhá položka
3. Třetí položka

```
<ol>
<li>První položka</li>
<li>Druhá položka</li>
<li>Třetí položka</li>
</ol>
```

Nečíslované seznamy

- První položka
- Druhá položka
- Třetí položka

```
<ul>
<li>První položka</li>
<li>Druhá položka</li>
<li>Třetí položka</li>
</ul>
```

Vnořování seznamů

- Seznam **** a **** může obsahovat jen elementy ****
- Položka seznamu **** může obsahovat libovolné blokové i řádkové elementy
- Je tedy možné i vnořovat seznamy

```
<ul>
<li>Jméno může být
    <ul>
        <li>Jan</li>
        <li>Jana</li>
    </ul>
</li>
<li> ... </li>
</ul>
```

Definiční seznamy

Jméno

Jan Novák

E-mail

jan@novak.cz

```
<dl>
<dt>Jméno</dt>
  <dd>Jan Novák</dd>
<dt>E-mail</dt>
  <dd>jan@novak.cz</dd>
</dl>
```

Položky definičního seznamu

- Seznam **<dl>** může obsahovat jen elementy **<dt>** a **<dd>** (v libovolné posloupnosti)
- Termín **<dt>** nemůže obsahovat sekce a nadpisy
- Definice **<dd>** může obsahovat libovolné blokové i řádkové elementy

Tabulky

- Definice tabulky **<table>**
- Nadpis **<caption>**
- Řádky **<tr>**
- Buňky **<td>, <th>**

Tabulky (pokrač.)

Příklad tabulky

Jméno	Příjmení
Jan	Novák
Josef	Dvořák

Tabulky (pokrač.)

```
<table>
<caption>Příklad tabulky</caption>
<tr>
    <th>Jméno</th>
    <th>Příjmení</th>
</tr><tr>
    <td>Jan</td>
    <td>Novák</td>
</tr><tr>
    <td>Josef</td>
    <td>Dvořák</td>
</tr>
</table>
```

Atributy buňek <th>, <td>

colspan=n	Šířka ve sloupcích
rowspan=n	Výška v řádcích

Colspan

```
<table>
<tr><th colspan="2">Jméno</th><th>Věk</th></tr>
<tr><td>Jan</td><td>Novák</td><td>23</td></tr>
<tr><td>Jiří</td><td>Dvořák</td><td>28</td></tr>
</table>
```

Jméno	Věk
Jan	23
Jiří	28

Rowspan

```
<table>
<tr><th rowspan="2">Jméno</th><th colspan="2">Průměr</th>
</tr>
<tr><th>2003</th><th>2004</th></tr>
<tr><td>Jan Novák</td><td>2.32</td><td>1.97</td></tr>
</table>
```

Jméno	Průměr	
	2003	2004
Jan Novák	2.32	1.97

Obrázky

```

```



- Obrázek je **řádkový** element
- Nepárová značka (element je vždy prázdný)

Rozměry obrázku

- Atributy **height** a **width**

```

```

- Nepovinné atributy, prohlížeč umí určit rozměry sám
- Obrázek musí být načten zvlášť – to trvá nějaký čas
- Je vhodné rozměry udávat – lepší renderování stránky
- Jsou-li zadány jiné hodnoty, obrázek je zvětšen nebo zmenšen

Formáty obrázků

- **Joint Photographic Experts Group JPEG (*.jpg)**
 - ztrátová komprese
 - vhodný pro uložení fotografií
- **Graphics Interchange Format (*.gif)**
 - bezztrátová komprese, průhlednost, animace
 - indexované barvy
 - zastaralý, již není důvod používat
- **Portable Network Graphics (*.png)**
 - náhrada gif, bez animací
- **Ostatní formáty** – velmi nízká podpora v prohlížečích

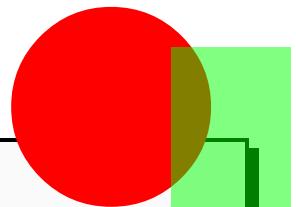
Vektorové formáty

- **SVG (Scalable Vector Graphics)**
 - Založeno na XML
 - Aktuální prohlížeče podporují nativně (IE až od verze 9)
 - Různé způsoby vložení do dokumentu: ``, `<object>`
 - HTML 5 umožňuje vložení SVG přímo do HTML
 - Značka `<svg>`

Vložené SVG

- Přímo v HTML dokumentu

```
<svg height="120"  
      xmlns="http://www.w3.org/2000/svg">  
  <circle cx="50" cy="50" r="50" fill="red" />  
  <rect x="80" y="20" height="60" width="60"  
        style="stroke:none; fill:#00ff00;  
        fill-opacity: 0.5;" />  
</svg>
```



Obrázek jako odkaz

Klikněte pro pokračování

```
<a href="slide0500">  
      
</a>
```

Klikněte pro pokračování



Obrázek jako odkaz (II)

Klikněte pro pokračování

```
<a href="slide0510">  
      
</a>
```

Klikněte pro pokračování



- Atribut style je možno nahradit centrální definicí stylu umístěnou jinde
 - viz. příští přednáška

Obrázek v textu (s popisem)

- Přiřazení popisu k obrázku (videu, ...)

```
<figure>
    
    <figcaption>Popisek obrázku</figcaption>
</figure>
```

Horizontální oddělovač

- Horizontální čára, logický oddělovač, změna tématu
 - Prázdný, bez atributů: <hr>
-

Globální atributy

- Atributy použitelné pro všechny elementy

title=text	Titulek elementu
lang=cs en ...	Jazyk obsahu
tabindex=0-n	Pořadí aktivace přes TAB
dir=rtl ltr	Směr psaní textu
hidden	Skrytý prvek
id=jmeno	Jednoznačný identifikátor
class=jmeno	Použití pro kaskádové styly (CSS)
style=styl	

Atribut title

- Dodatečná informace o elementu
- V dokumentu se nezobrazí, při přejetí myší se zobrazí v „bublině“
- Užitečné pro zápis poznámek a ve spojení s odkazy

```
<a href="..." title="Klikněte pro další slide">odkazy</a>
```

Generické elementy

- Dva obecné elementy:
 - **** - obecný řádkový element
 - **<div>** - obecný blokový element
- Vymezují část řádku nebo blok s blíže neurčeným speciálním významem
 - Blížší určení lze doplnit např. pomocí atributů **class** nebo **title**
- Obvykle pro definici vzhledu v kombinaci s CSS

Odkazy

- WWW Consortium – <http://www.w3.org/>
- Specifikace HTML 5 – www.w3.org/TR/html5/
- HTML Validator – <http://validator.w3.org/>
- SVG – <http://www.w3.org/Graphics/SVG/>
- Jak psát web – <http://www.jakpsatweb.cz/>

3. Kaskádové styly (CSS)

Jemný úvod

Vizuální styl v HTML

- Původní stav:
 - Specifikace vizuální prezentace společně s obsahem v HTML kódu
- Nesnadno udržovatelné
 - Nepřehledné
 - Nejednotný způsob definice (různé atributy)
 - Styl dokumentu je „rozprostřen“ v celém dokumentu
 - Změna vzhledu webu znamená změnit všechny dokumenty
- Dynamicky generovaný obsah
 - Program musí generovat obsah i vzhled
 - Při změně vzhledu nutná změna programu

Návrh řešení

- Odebrat prostředky pro definici vzhledu z HTML
 - V HTML 5 již nejsou značky definující vizuální styl
 - Důraz kladen na **sémantiku** každého elementu
- Vytvořit oddělené prostředky pro definici vzhledu
 - Cascading Stylesheets (CSS)

Kaskádové styly

- Definují vzhled dokumentu nezávisle na jeho obsahu
- HTML dokument definuje pouze strukturu a obsah dokumentu
- Vzhled je definován odděleně
 - Snadno modifikovatelný
 - Může být definován centrálně pro celou množinu stránek
 - Konzistentní vzhled stránek
 - Snadna tvorba nové nebo alternativní verze vzhledu
 - Jednotný způsob definice stylu pro všechny elementy
 - Jednoznačné a přehledné

Verze CSS

- Cascading Style Sheets, 1996 (CSS 1)
 - Neúplná podpora v Netscape4 a IE4
 - Problémy s kompatibilitou
 - Slušná podpora v Netscape5, Mozilla, IE5.x
- CSS 2 (1998), CSS2.1 (2011 – 2014)
 - Kompletní podpora ve všech moderních prohlížečích
- CSS 2.2 (dosud udržováno)
 - Opravy chyb a zpřesnění specifikace
- CSS 3 - ve fázi návrhu
 - Části již jsou dokončeny (postup prací)

Style Sheet

- Style sheet = stylový předpis
- Souhrn pravidel určujících styl dokumentu
- Každý dokument je popsán stylesheety ze tří zdrojů:
 1. Styl připravený tvůrcem stránek
 2. Uživatelský styl (vlastní definice uživatele)
 3. Styl prohlížeče
- Každý styl obsahuje pravidla týkající se různých elementů
 - Typicky více pravidel pro každý element
- Relevantní pravidla se seřadí do **kaskády** a kompletně definují výsledný vzhled každého elementu

Připojení stylu k HTML dokumentu

```
<!DOCTYPE html>
<html>
<head>
    <title>Titulek</title>
    <link rel="stylesheet" type="text/css"
          href="styl.css">
</head>
<body>
    ...

```

Existuje více možností, podrobněji později.

CSS Syntax

- Stylový předpis určuje vizuální (a jiné) vlastnosti jednotlivých elementů v HTML kódu
- Stylový předpis je posloupnost ***pravidel***:

```
selektor { deklarace vlastností }
selektor { deklarace vlastností }
...
...
```

- **Selektor** určuje, pro které elementy dané pravidlo platí
- **Vlastnosti** definují vizuální (případně další) vlastnosti dotyčných elementů

Deklarace vlastností

- Deklarujeme hodnoty jednotlivých vlastností
- Tvar definice:
 - **vlastnost: hodnota; vlastnost: hodnota; ...**
- Specifikace CSS definuje pevnou množinu existujících vlastností
- Každá vlastnost má **jméno**
 - Např. **color, font-size**
- Jsou definovány **možné hodnoty** každé vlastnosti
- Např:
 - **color: blue; font-size: 12pt;**

Selektory

- * (nebo prázdný selektor)
 - Pravidlo platí pro všechny elementy
 - `* { color: blue; }`
- Jméno elementu
 - Pravidlo platí pro všechny elementy daného jména
 - `h2 { color: green; }`

Selektory

- Třída elementu
 - Pravidlo platí pro elementy s danou hodnotou v atributu **class**
 - Název třídy je uvozen znakem „.”
 - **.dulezite { color: red; }**
 - ***.dulezite { color: red; }**
 - **h2.vyrazne { color: red; }**
- Identifikátor elementu
 - Pravidlo platí pro elementy s danou hodnotou atributu **id**
 - Identifikátor předchází „#”
 - **#menu { color: red; }**

Třídy elementů

- Třída = množina elementů v dokumentu
- Libovolný počet tříd elementů
- Každá třída má jméno
 - Vhodně zvolí tvůrce stránky
- Element náleží do nějaké třídy, pokud jeho atribut **class** obsahuje jméno této třídy
- Element může náležet do více tříd
 - Jména tříd v attributu **class** jsou oddělena mezerami
- V jedné třídě může být libovolný počet elementů

Třídy elementů – příklad

```
<h1 class="hlavni">....</h1>

<div class="dulezite">
    ...
</div>

<div class="dulezite vyrazne">
    ...
</div>
```

Identifikátory elementů

- Každému elementu lze přiřadit identifikátor pomocí atributu **id**:

```
<h1 id="titulek">....</h1>  
  
<div id="obsah">  
    ...  
</div>
```

- Identifikátor musí být **jedinečný** v rámci dokumentu

Příklad

- Pravidlo: **span { }**

```
<h1 id="nadpis" class="red n1">Můj dokument</h1>

<p class="n1">
Potřebujeme <span class="a1">nastavit</span>
různé <span class="b1">styly</span> pro všemožné
elementy v <span class="a1 b1">dokumentu</span>.
</p>
```

Příklad

- Pravidlo: .n1 { }

```
<h1 id="nadpis" class="red n1">Můj dokument</h1>
```

```
<p class="n1">  
Potřebujeme <span class="a1">nastavit</span>  
různé <span class="b1">styly</span> pro všemožné  
elementy v <span class="a1 b1">dokumentu</span>.  
</p>
```

Příklad

- Pravidlo: p.n1 { }

```
<h1 id="nadpis" class="red n1">Můj dokument</h1>
```

```
<p class="n1">  
Potřebujeme <span class="a1">nastavit</span>  
různé <span class="b1">styly</span> pro všemožné  
elementy v <span class="a1 b1">dokumentu</span>.  
</p>
```

Příklad

- Pravidlo: `.a1 { }`

```
<h1 id="nadpis" class="red n1">Můj dokument</h1>

<p class="n1">
Potřebujeme <span class="a1">nastavit</span>
různé <span class="b1">styly</span> pro všemožné
elementy v <span class="a1 b1">dokumentu</span>.
</p>
```

Příklad

- Pravidlo: .b1 { }

```
<h1 id="nadpis" class="red n1">Můj dokument</h1>

<p class="n1">
Potřebujeme <span class="a1">nastavit</span>
různé <span class="b1">styly</span> pro všemožné
elementy v <span class="a1 b1">dokumentu</span>.
</p>
```

Příklad

- Pravidlo: #nadpis { }

```
<h1 id="nadpis" class="red n1">Můj dokument</h1>
```

```
<p class="n1">  
Potřebujeme <span class="a1">nastavit</span>  
různé <span class="b1">styly</span> pro všemožné  
elementy v <span class="a1 b1">dokumentu</span>.  
</p>
```

Příklad

- Pravidlo: * { }

```
<h1 id="nadpis" class="red n1">Můj dokument</h1>
```

```
<p class="n1">  
Potřebujeme <span class="a1">nastavit</span>  
různé <span class="b1">styly</span> pro všemožné  
elementy v <span class="a1 b1">dokumentu</span>.  
</p>
```

Strukturované selektory

- Vychází ze stromové struktury HTML dokumentu
- Čteme nejlépe odzadu
- **s1 s2 { }**
 - Element odpovídající s2, který je **potomkem** elementu s1
 - Např. `.vyrazne strong { }`
- **s1 > s2 { }**
 - Element odpovídající s2, který je **synovským uzlem** elementu s1
 - Např. `a > em { }`
- **s1 + s2 { }**
 - Element odpovídající s2, který je **předcházen** elementem s1
 - Např. `table + p { }`

Selektory s atributy

- **značka[atribut] { }**
 - Značka s nastaveným atributem **atribut**
 - Např. **p[class]** { }
- **značka[atribut="hodnota"] { }**
 - Značka s atributem **atribut** nastaveným na hodnotu **hodnota**
 - Např. **p[class]** { }
- **značka[atribut~="hodnota"] { }**
 - Značka s atributem **atribut**, který obsahuje hodnoty oddělené mezerami, z nichž jedna je rovna **hodnota**
 - Např. **p[class~="red"]** { }
- **značka[atribut|= "hodnota"] { }**
 - Hodnota začíná daným řetězcem nebo za řetězcem následuje pomlčka.

Seskupování

- Více pravidel pro stejný selektor

```
h1 { color: red; }  
h1 { font-size: 20pt; }  
h2 { color: red; font-size: 18pt; }
```

- Více selektorů pro jedno pravidlo

```
h1, h2, h3 { color: red; }
```

Hodnoty vlastností

- Čísla
 - Celá i desetinná (např. **6.2**)
 - Často v omezeném rozsahu (např. nezáporná)
- Délky
 - V různých délkových jednotkách
- Procentní vyjádření
 - Velikost nebo pozice (např. **12.5%**)
 - Interpretace závisí na kontextu
- URI
 - `url('http://www.zdroj.cz/obrazek.png')`
 - Absolutní i relativní

Hodnoty vlastností

- Barvy
 - Popředí (písmo, rámečky)
 - Pozadí
- Speciální hodnoty
 - Typické pro jednotlivé vlastnosti

Délky v CSS

- U různých vlastností v CSS - šířky, délky, pozice, ...
- Zápis číslem a jednotkou, např. **10mm** (bez mezery)
- Absolutní jednotky:
 - pixely - **px**
 - milimetry - **mm**
 - centimetry - **cm**
 - palce - **in** (1in = 2.54cm)
 - typografické body, pica - **pt, pc** (1pc = 12pt, 1pt = 1/72in)
- Relativní jednotky
 - čtverčík (eM) - **em** - šířka velkého M
 - x-height - **ex** - výška malého x

Barvy v CSS

- Pojmenované barvy
 - maroon, red, orange, yellow, olive
purple, fuchsia, white, lime, green
navy, blue, aqua, teal
black, silver, gray
 - CSS3 zavádí rozšířenou sadu barev
- RGB model - různé způsoby zápisu
 - `rgb(100%,50%,20%)` **Ukázka**
 - `rgb(255,10,127)` **Ukázka**
 - `#a0ff00` **Ukázka**
 - `#afb` **Ukázka**
- Barva textu:
 - **color: #aa7f5b**

Pozadí elementu

- Barva pozadí
 - **background-color: transparent**
 - **background-color: red**
- Obrázek na pozadí
 - **background-image: none**
 - **background-image: url('abstract.png')**
- Opakování obrázku
 - **background-repeat: repeat**
 - **background-repeat: no-repeat**
 - **background-repeat: repeat-x**
 - **background-repeat: repeat-y**

Pozadí elementu (II)

- Pozice obrázku na pozadí (v rámci elementu)
 - **background-position:** 20% 5% (hor., vert.)
 - **background-position:** 2cm 1cm
 - **background-position:** top left | center right
 - **background-position:** center
- Chování při scrollování
 - **background-attachment:** scroll
 - **background-attachment:** fixed

Vlastnost background

- Shrnuje všechny vlastnosti **background-xxxx**

```
background: red url('pozadi5.gif')  
           top left no-repeat;
```

- Neuvedené vlastnosti opět nastavuje na implicitní hodnoty

Rámečky

- Tloušťka rámečku
 - **border-width: 2px**
 - **border-top-width: 2px**
 - **border-right-width: 2px**
 - **border-bottom-width: 2px**
 - **border-left-width: 2px**
- Barva rámečku
 - **border-color: red**

Rámečky (II)

- Styl rámečku
 - **border-style: none**
 - **border-style: solid**
 - **border-style: dotted**
 - **border-style: dashed**
 - **border-style: double**
 - **border-style: groove**
 - **border-style: ridge**
 - **border-style: inset**
 - **border-style: outset**

Shrnující zápis

- Pro celý rámeček

```
border: 2px solid yellow;
```

- Pro jednotlivé strany

```
border-top: 2px solid yellow;  
border-bottom: 3px dotted red;  
border-left: 1px dashed #ffa;  
border-right: 2px double #ffaaff;
```

Druh písma (font)

- Vlastnost **font-family**:
- Pozor: font musí být k dispozici v prohlížeči
- Generické druhy písma (k dispozici všude)
 - serif
 - sans-serif
 - monotype
- Ostatní použitelné:
 - Times New Roman, Times, Arial, Helvetica, **Verdana**, Courier New a Courier

Druh písma (II)

- Je nutné vždy specifikovat alternativy
- Pokud název obsahuje mezery, musí být v úvozovkách
- Poslední alternativa musí být generická

```
p.c1 { font-family: Verdana, Arial, sans-serif; }  
p.c2 { font-family: 'Times New Roman', Times, serif; }  
p.c3 { font-family: 'Courier New', Courier, monospace; }
```

Vlastnosti písma

- Styl písma
 - **font-style:** normal
 - **font-style:** *italic*
 - **font-style:** *oblique*
- Váha písma
 - **font-weight:** normal
 - **font-weight:** bold
- Varianta
 - **font-variant:** normal
 - **font-variant:** SMALL-CAPS

Velikost písma

- Absolutní velikost
 - **font-size: 12px**
- Relativní velikost
 - **font-size: 2em**
- Procenta
 - **font-size: 150%**
- Slovní vyjádření
 - **small, x-small, xx-small**
 - **medium**
 - **large, x-large, xx-large**

Vlastnost font

- Shrnuje některé vlastnosti **font-xxxx**:
 - Váha a styl
 - Velikost
 - Řez písma
- Nejprve se všechny hodnoty nastaví na počáteční
- Poté se změní podle nastavených hodnot

```
body { font: bold 20px sans-serif; }
body { font-weight: bold; font-style: normal;
       font-size: 20px; font-family: sans-serif; }
```

Vlastnosti textu

- Dekorace textu
 - **text-decoration:** none
 - **text-decoration:** overline
 - **text-decoration:** underline
 - **text-decoration:** ~~line-through~~
- Transformace textu
 - **text-transform:** none
 - **text-transform:** Capitalize
 - **text-transform:** UPPERCASE
 - **text-transform:** lowercase

Zarovnávání

- Horizontální zarovnávání
 - **text-align: left**
 - **text-align: right**
 - **text-align: center**
 - **text-align: justify**
- Vertikální zarovnávání
 - **vertical-align: baseline**
 - **vertical-align: sub**
 - **vertical-align: super**
 - **vertical-align: top**
 - **vertical-align: bottom**
 - **vertical-align: 30%**

Mezery

- Mezera mezi slovy
 - **word-spacing: 50px**
- Mezera mezi písmeny
 - **letter-spacing: 5px**
- Výška řádku
 - **line-height: 150%**

První řádek

Druhý řádek

Třetí řádek

Čtvrtý řádek

Zobrazení

- Forma zobrazení
 - **display: block**
 - **display: inline**
 - **display: none**
- Zpracování mezer
 - **white-space: normal**
 - **white-space: pre**
 - **white-space: nowrap**

Odkazy

- Specifikace CSS: <http://www.w3.org/Style/CSS/current-work>
- CSS Zen Garden: <http://www.csszengarden.com/>

4. Kaskádové styly (II)

Základní mechanismy CSS

Připojení stylů k HTML dokumentu

- Tři možnosti připojení stylů:
 1. Externí stylesheet (nejběžnější)
 2. Deklarace v hlavičce dokumentu
 3. Inline styly

Externí stylesheet

```
<!DOCTYPE html>
<html>
<head>
    <title>Titulek</title>
    <link rel="stylesheet" type="text/css"
          href="style.css">
</head>
<body>
    ...

```

style.css

```
body { color: white; }
h1 { color: red; }
```

Deklarace stylu v hlavičce

```
<!DOCTYPE html>
<html>
<head>
    <title>Titulek</title>
    <style type="text/css">
        body { color: white; }
        h1 { color: red; }
    </style>
</head>
<body>
    ...

```

- Nevýhoda: nutno definovat v každém dokumentu – pracné udržování konzistence

Inline styly

```
...
</head>
<body>
    <h1 style="color: red">Nadpis</h1>
    <span style="font-weight: bold">Tučné písmo</div>
    <div style="font-size: 20px">
        Blok velkým písmem
    </div>
    ...
</body>
```

- Uvádíme pouze deklaraci vlastností bez selektoru a { }

Postup definice stylu

- Definice základního vzhledu elementů (nadpisů, odstavců, tabulek, ...)
 - Zabudovaný styl prohlížeče
 - Dodaný styl na základě grafického návrhu
- Specifikace detailů a specifických případů
 - Rozmístění konkrétních prvků (záhlaví, ...)
 - Speciální případy (tabulek, nadpisů, ...)
- Využijeme mechanismy CSS
 - Kaskáda pravidel
 - Dědičnost

Postup aplikace pravidel

- Jednomu elementu může odpovídat více selektorů

```
<p class="uvod">Následující text je  
  <span class="dulezite">důležitý</span></p>
```

```
p.uvod span { color: red; }  
p span { color: blue; font-weight: bold; }
```

- Jakou barvu bude mít slovo „důležitý“ ?

Direktiva !important

- Vlastnost lze označit jako důležitou pomocí direktivy **!important**
- Např. **table { color: blue !important }**
- Takto deklarovaná vlastnost má pak přednost před ostatními deklaracemi stejné vlastnosti

Kaskáda pravidel

- Vezmeme všechna pravidla, jejichž selektor odpovídá danému elementu
- Pokud je některá vlastnost definována opakovaně, vybereme hodnotu podle následujících kritérií:
 1. Zdroj pravidla
 - Různé zdroje jsou různě významné
 2. Seřadíme podle specifičnosti selektoru
 - Čím konkrétnější selektor, tím větší prioritu má pravidlo
 3. Seřadíme podle pořadí specifikace
 - Má-li pravidlo stejný zdroj, prioritu a specifičnost pak vyhrává ***později*** uvedené pravidlo

Zdroj pravidla

- Význam jednotlivých pravidel podle zdroje od nejméně významného k nejvíce významnému je stanoven takto:
 1. Styly prohlížeče
 2. Normální styly uživatele
 3. Normální styly autora dokumentu
 4. Prioritní (!important) styly autora dokumentu
 5. Prioritní styly uživatele
- Autor stránek tedy může uvažovat pouze 3 a 4

Specifičnost

- Specifičnost je číslo **abcd**, kde
 - **a=1** pokud se jedná o inline styl, jinak **a=0**
 - **b** je počet atributů **id** v selektoru
 - **c** je počet tříd v selektoru
 - **d** je počet názvů elementů v selektoru
- **p.uvod span { }:** 0 - 0 - 1 - 2 ~ **12**
- **p span { }:** 0 - 0 - 0 - 2 ~ **2**

Zpět k otázce

- Jakou barvu bude mít slovo „důležitý“ ?

```
<p class="uvod">Následující text je  
<span class="dulezite">důležitý</span></p>
```

```
p.uvod span { color: red; }  
p span { color: blue; font-weight: bold; }
```

- Zvítězí specifičtější pravidlo => ?

Zpět k otázce (II)

- Jakou barvu bude mít slovo „důležitý“ ?

```
<p class="zaver">Následující text je  
<span class="dulezite">důležitý</span></p>
```

```
p.uvod span { color: red; }  
p span { color: blue; font-weight: bold; }
```

- Pouze jeden kandidát => ?

Zpět k otázce (III)

- Jakou barvu bude mít slovo „důležitý“ ?

```
<p class="uvod zaver">Následující text je  
<span class="dulezite">důležitý</span></p>
```

```
p.uvod span { color: red; }  
p.zaver span { color: blue; font-weight: bold; }
```

- Rozhodne pořadí specifikace => ?

Dědičnost vlastností

- Hodnoty některých vlastností se dědí na potomky

```
<p class="uvod zaver">Následující text je  
  <span class="dulezite">důležitý</span></p>
```

```
p { color: red; border: 1px solid blue; }
```

- Barva textu se projeví i v synovském prvku
- Rámeček bude mít pouze otcovský prvek `<p>`
- O tom, které vlastnosti se dědí rozhoduje specifikace CSS

Běžný případ použití dědičnosti

```
body {  
    color: white;  
    background-color: black;  
}  
  
h1 {  
    color: red;  
}
```

- Veškerý text dokumentu bude psán bílou barvou
 - Všechny elementy zdědí barvu od kořenového elementu
- Pouze nadpisu úrovně 1 (a vnořené elementy) budou červené
 - Pro h1 jsme definovali barvu

Hodnota `inherit`

- Každé vlastnosti lze přiřadit speciální hodnota **`inherit`**
- Hodnota této vlastnosti se pak vždy zdědí od rodičovského prvku

```
<div id="menu">
    <h1>Menu</h1>
    <p>První odstavec</p>
    <p>Druhý odstavec</p>
</div>
```

Hodnota inherit - Příklad

Menu

První odstavec

Druhý odstavec

```
#menu { border: 3px #99ff99 solid; }
```

Hodnota inherit - Příklad

Menu

První odstavec

Druhý odstavec

```
#menu { border: 3px #99ff99 solid; }
#menu p { border: inherit; }
```

Hodnota inherit - Příklad

Menu

První odstavec

Druhý odstavec

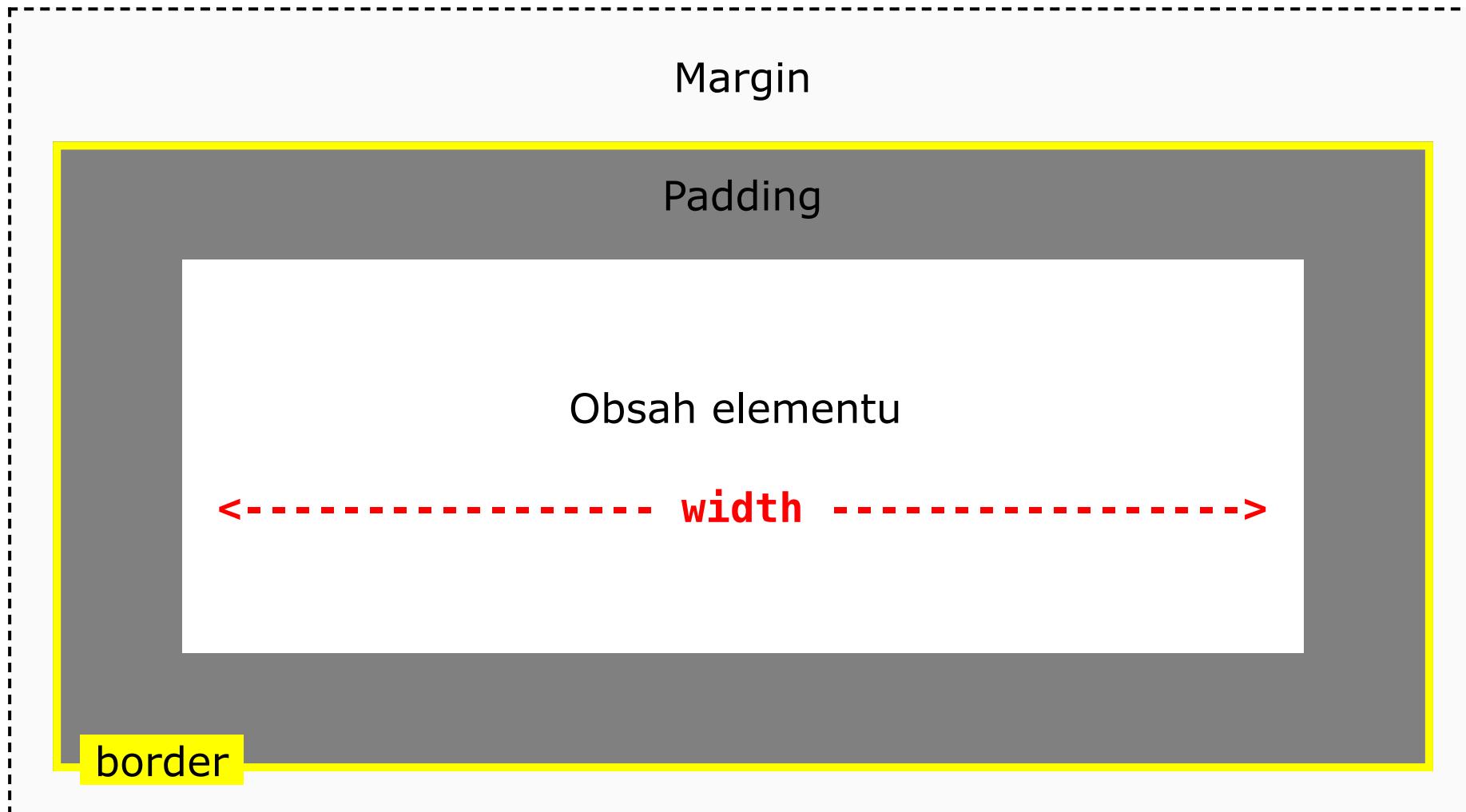
```
#menu { border: 10px #99ff99 solid; }
#menu p { border-style: dashed; border-color: inherit; }
```

Práce s blokovými elementy

Box model

- Model popisující rozměry libovolného objektu na stránce
- Složky box modelu:
 - Šířka a výška obsahu
 - Tloušťka rámečku
 - Okraje – vzdálenost od okolních prvků
- Každou složku lze nastavit pomocí příslušných CSS vlastností

Box model



Výška a šířka obsahu

- U **řádkových** elementů dána šířkou a výškou textu
- U blokových elementů může být
 - Zadána pomocí vlastností **width** a **height**
 - Vypočtena na základě ostatních vlastností:
 - Velikost nadřazeného elementu
 - Velikost obsahu
 - Okraje a rámečky
- Kořenový element (**<html>**)
 - Rozměry ovlivněny velikostí okna prohlížeče

Šířka a výška obsahu

- Šířka
 - **width: auto**
 - **width: 120px**
 - **width: 60%**
- Výška
 - **height: auto**
 - **height: 30px**
 - **height: 80%**

Minimální šířka a výška

- Maximální rozměry
 - **max-width: 20em;**
 - **max-height: 50em;**
- Minimální rozměry
 - **min-width: 20em;**
 - **min-height: 50em;**
- Postup aplikace:
 - **width -> max-width -> min-width**

Vnější okraj (margin)

- Po jednotlivých stranách
 - **margin-top:**, **margin-bottom:**
 - **margin-left:**, **margin-right:**
- Najednou
 - **margin: 2em 1em 3em 2em** - top, right, bottom, left
 - **margin: 2em 1em 1em** - top, right&left, bottom
 - **margin: 2em 1.5em** - top&bottom, right&left
 - **margin: 1em** - všechny

Vnitřní okraj (padding)

- Po jednotlivých stranách
 - **padding-top:**, **padding-bottom:**
 - **padding-left:**, **padding-right:**
- Najednou
 - **padding: 2em 1em 3em 2em** - top, right, bottom, left
 - **padding: 2em 1em 1em** - top, right&left, bottom
 - **padding: 2em 1.5em** - top&bottom, right&left
 - **padding: 1em** - všechny

Hodnoty auto

- Vlastnosti **margin**, **width** a **height** mohou mít zvláštní hodnotu **auto**
 - Pro **width** a **height** výchozí
 - U **margin** je výchozí hodnota 0
- Skutečné hodnoty vlastností s hodnotou **auto** se dopočítají podle ostatních hodnot
- Způsob výpočtu se liší podle způsobu rozmístění objektů

Způsob rozmístění objektů

- Normální tok (normal flow)
 - Tzv. ***in-flow*** elementy
 - Elementy se rozmísťují v pořadí jak jsou uvedeny v HTML kódu
 - Řádkové elementy na řádek, blokové pod sebe
- Plovoucí bloky (floating blocks)
 - Elementy obtékané in-flow obsahem následujícím po nich
 - Vlastnost **float**
- Pozicované bloky
 - Pozice bloku na stránce je explicitně určena
 - Vlastnost **position**

Šířka bloku – normální tok

- **Blok vždy zabírá celou šířku obsahu bloku, ve kterém je obsažen**
 - U kořenového elementu celou šířku obrazovky
- Platí tedy:
margin-left + border-left-width +
+ padding-left + width + padding-right +
+ border-right-width + margin-right =
šířka_obsahujícího_bloku
- Z toho lze dopočítat případné hodnoty **auto**
- Pokud žádná hodnota není **auto**, je jedna hodnota nadbytečná
=> ignoruje se hodnota **pravého okraje**

Výpočet hodnot auto

- Pokud je pouze jedna hodnota **auto**, dopočte se z předchozí rovnice
- Pokud šířka není zadána (**width=auto**), hodnoty **auto** u okrajů se interpretují jako **0**
- Pokud levý i pravý okraj mají hodnotu **auto**, jejich výsledné velikosti jsou stejné => **centrování bloku v obsahujícím bloku**

Výška bloku

- Je-li **height=auto**, určí se výška tak, aby se vešly všechny obsažené elementy
 - Standardně se uvažují jen in-flow elementy
 - Lze ovlivnit nastavením **overflow**
- U horního a spodního okraje se hodnota **auto** považuje za **0**

Spojování okrajů

- **Horizontální** okraje se **nikdy nespojují**
- **Vertikální** okraje se spojují
 - U dvou pod sebou ležících **neplovoucích** bloků
 - U dvou vzájemně vnořených bloků (horní nebo dolní okraj)
 - Horní okraj pouze pokud vnitřní objekt má **clear: none**
 - Výsledný okraj je **maximem** spojovaných okrajů
- Příklad

Ořezávání

- Vlastnost **overflow** - co dělat když se text nevejde do rámečku.
 - **overflow: visible** - text přeteče (implicitní)
 - **overflow: hidden** - oříznutí
 - **overflow: scroll** - rolování
 - **overflow: auto** - rolování, je-li třeba

```
<div style="width: 5em; height: 6em;  
border: 2px solid black;">  
Rámeček obsahující dlouhý text.  
</div>
```

Ořezávání - příklad

overflow: visible

Rámeček
obsahující
doooooooooouhý
text.

overflow: hidden

Rámeček
obsahující
text.

overflow: scroll

Rámeček
obsahující
text.

overflow: auto

Rámeček
obsahující
text.

5. Kaskádové styly (III)

Rozmístování bloků na stránce

Šířka a výška

- Vlastnosti **width** a **height** určují šířku/výšku boxu
 - Hodnota **auto** nebo zadaná délka
- Vlastnost **box-sizing** určuje, co daný rozměr zahrnuje
 - **box-sizing: content-box** – **width** a **height** nastavují rozměry **obsahu**
 - **box-sizing: border-box** – **width** a **height** nastavují rozměry **včetně border a padding**
- **box-sizing** se nedědí, je třeba nastavit pro všechny dotčené boxy. Např:

```
.column div {  
    box-sizing: border-box;  
}
```

box-sizing – příklad

- **box-sizing: content-box**

Box 1

width: 15em

padding: 0

Box 1

width: 15em

padding: 1em

Box 1

width: 13em

padding: 1em

box-sizing – příklad

- **box-sizing: border-box**

Box 1

width: 15em

padding: 0

Box 1

width: 15em

padding: 1em

Box 1

width: 13em

padding: 1em

Způsob rozmístění objektů

- Normální tok (normal flow)
 - Tzv. ***in-flow*** elementy
 - Elementy se rozmísťují v pořadí jak jsou uvedeny v HTML kódu
 - Řádkové elementy na řádek, blokové pod sebe
- Plovoucí bloky (floating blocks)
 - Elementy obtékané in-flow obsahem následujícím po nich
 - Vlastnost **float**
- Pozicované bloky
 - Pozice bloku na stránce je explicitně určena
 - Vlastnost **position**

Plovoucí objekty

- Objekty obtékané okolním obsahem (zleva nebo zprava)
- Vytvoření polovoucího objektu:
 - **float: left**
 - **float: right**
- Plovoucí elementy se stávají automaticky **blokovými**
- Plovoucí objekt je
 - Objekt je vyjmut z normálního toku (nijak se neprojeví na původním místě)
 - Odsunut ke kraji rodičovského prvku
- Mění se postup výpočtu šířky
 - Nejlépe je mít nastavenou šířku
 - Pokud není, určí se automaticky

Ukázka plovoucích objektů

- Výchozí stav

```
...
<div id="menu" class="plovouci">
    <h1>Menu</h1>
    <ul>
        <li>Expozice</li>
        <li>Kolize</li>
        <li>Krize</li>
        <li>Peripetie</li>
        <li>Katastrofa</li>
    </ul>
</div>
...
```

Vytvoření plovoucího objektu

```
<style type="text/css">
    #menu { background-color: #eef;
            border: 1px solid #aaf;
            width: 12em; }
    .plovouci { float: left; }
</style>
```

Ukázka

Okraje

```
<style type="text/css">
    #menu { background-color: #eef;
            border: 1px solid #aaf;
            width: 12em;
            padding: 0;
            margin: 0 1em; }
    .plovouci { float: left; }
</style>
```

Nadpis

```
<style type="text/css">
    #menu { ... }
    #menu h1 { font-weight: bold;
                font-size: 100%;
                color: #fff;
                background-color: #55f;
                margin: 0;
                padding: 0.5em; }
    .plovouci { float: left; }
</style>
```

Ukázka

Velikost bloku odstavce

```
<style type="text/css">
    #menu { ... }
    #menu h1 { font-weight: bold;
                font-size: 100%;
                color: #fff;
                background-color: #55f;
                margin: 0;
                padding: 0.5em; }
    .plovouci { float: left; }
    p { border: 5px solid red; }
</style>
```

Ukázka

Potlačení obtékání

- Plovoucí objekty umístuje prohlížeč
 - Vedle sebe na příslušný okraj
 - Není-li místo tak pod sebe
 - => Plovoucí objekt může přesáhnout rozměr rodičovského objektu
- Pokud některý text nemá obtékat plovoucí objekty:
 - **clear: left | right | both**
- Čeká se na umístění předchozích plovoucích objektů
- Příklad

Výška obsahujícího bloku (opět)

- Má-li blok nastaveno **height: auto**, jeho výška se dopočítá podle obsahu
- Při **overflow: visible**
 - Uvažují se pouze in-flow elementy obsahu
 - Obsahuje-li element pouze plovoucí bloky, má ve výsledku **nulovou výšku**
- Jinak (např. **overflow: hidden**)
 - Uvažují se i plovoucí elementy
- Příklad

Výška bloku: příklad

```
<div class="obsah">
    <div class="sloupec">Sloupec 1</div>
    <div class="sloupec">Sloupec 2</div>
</div> <p>Další obsah</p>
```

```
.obsah {
    border: 5px solid blue;
}
.sloupec {
    float: left;
    background: gray;
}
```

Výška bloku: příklad (II)

```
.obsah {  
    overflow: visible;  
}
```

Sloupec 1 Sloupec 2 Další obsah

Výška bloku: příklad (III)

```
.obsah {  
    overflow: hidden;  
}
```

Sloupec 1 Sloupec 2

Další obsah

Pozicování

- Ruční nastavení polohy objektu na stránce
- Absolutní
 - Umístění objektu na konkrétní místo stránky nezávisle na ostatním textu
- Relativní
 - Posunutí oproti normální poloze objektu
- Vlastnost **position**
 - **position: static** – normální stav
 - **position: absolute** – absolutní pozicování
 - **position: relative** – relativní pozicování
 - **position: fixed** – fixní pozice

Relativní pozicování

- Poloha objektu oproti normálnímu umístění
 - **position: relative**
- Blok zůstává v normálním toku, pouze se zobrazí posunutý
- Vzdálenost od levého horního rohu:
 - **top: délka** - posunutí dolů
 - **left: délka** - posunutí doprava
- Vzdálenost od pravého dolního rohu:
 - **bottom: délka** - posunutí nahoru
 - **right: délka** - posunutí doleva
- Lze kombinovat
- Záporné hodnoty - opačný směr posunutí

Relativní pozicování - příklad

```
<style>
    .slovo {
        font-weight: bold;
    }
</style>
...
Toto je <span class="slovo">tučný</span> text.
```

Toto je **tučný** text.

Relativní pozicování - příklad

```
<style>
    .slovo {
        font-weight: bold;
        position: relative;
        top: -0.5em;
        left: 5px;
    }
</style>
...
Toto je <span class="slovo">tučný</span> text.
```

Toto je **tučný**text.

Absolutní pozicování

- Objekt je **vyjmut** z normálního toku
 - Neprojeví se v původním umístění
- Je umístěn na absolutní pozici na stránce
 - Může překrývat objekty, které tam již jsou
 - Je nutné mu „udělat místo“
- Vzdálenost od levého horního rohu **obsahujícího bloku**
 - **top: délka**
 - **left: délka**
- Od pravého dolního rohu
 - **bottom: délka**
 - **right: délka**

Obsahující blok

- Počáteční obsahující blok: **Viewport**
- Elementy s pozicováním **relative** nebo **static**
 - Obsahující blok je nejbližší blokový předek.
- Elementy s pozicováním **absolute**
 - Nejbližší blokový předek s pozicováním **relative**, **absolute** nebo **fixed**
- Elementy s pozicováním **fixed**
 - Obsahující blok viewport.

Souřadný systém

- Objekty, které mají **position: absolute, relative** nebo **fixed**, vytvářejí **vlastní souřadný systém**
 - Absolutně pozicované vnořené objekty se umístují pouze v rámci tohoto elementu
 - Nejjednodušší vytvoření části stránky s absolutně pozicovaným vnitřkem:

```
<div style="position: relative">  
...  
</div>
```

Absolutní pozicování - příklad

```
<style>                                tučný
    .slovo {
        font-weight: bold;
        position: absolute;
        top: 150px;
        left: 500px;
    }
</style>
...
Toto je <span class="slovo">tučný</span> text.
```

Toto je text.

Fixní pozicování

```
<style>                                tučný
    .slovo {
        font-weight: bold;
        position: fixed;
        top: 150px;
        left: 500px;
    }
</style>
...
Toto je <span class="slovo">tučný</span> text.
```

Toto je text.

Rozměry pozicovaných elementů

- Platí rovnice:
 $\text{left} + \text{margin-left} + \text{border-left-width} +$
 $+ \text{padding-left} + \text{width} + \text{padding-right} +$
 $+ \text{border-right-width} + \text{margin-right} + \text{right} =$
 $\text{šířka obsahujícího bloku}$
- Hodnoty **auto** pro **left** a **top** vycházejí ze **statické pozice**
- Pravidla pro dopočítávání hodnot ve specifikaci
- Analogicky pro výšku

Překrývání objektů

- Dva objekty se mohou překrývat (relativní i absolutní pozicování)
- Normální stav: později uvedený objekt překrývá dříve uvedený objekt
- Lze změnit pomocí **z-index**
 - **z-index: 10**
 - Číslo udává Z souřadnici objektu (čím vyšší tím „výše“ se objekt nachází)
 - Pokud není uvedeno, bere se **0**
 - Platí jen pro pozicované elementy

Překrývání: příklad

```
<div style="height: 4em; position: relative">
  <div class="ramecek"
    style="position: absolute;
    top: 30px; left: 180px">
    První rámeček</div>
  <div class="ramecek"
    style="position: absolute; top: 60px; left: 200px">
    Druhý rámeček</div>
</div>
```



Překrývání: příklad

```
<div style="height: 4em; position: relative">
  <div class="ramecek"
    style="position: absolute;
    top: 30px; left: 180px; z-index: 1">
    První rámeček</div>
  <div class="ramecek"
    style="position: absolute; top: 60px; left: 200px">
    Druhý rámeček</div>
</div>
```



Viditelnost objektů

- Dvě možnosti
 1. **display: none**
 - Prvek se při renderování zcela ignoruje
 2. **visibility: hidden**
 - Prvek zabírá místo na stránce, jako by byl zobrazen, ale nezobrazí se

Viditelnost – příklad

Toto je neviditelný text.

1. display: none

Toto je text.

2. visibility: hidden

Toto je text.

Proč skrývat text

1. **display: none**

- Prvky, které se zobrazí pouze v prohlížečích bez stylů (alternativní navigace, pomocné popisy, ...)
- Prvky určené pro jiná média (např. pro tisk)

2. **visibility: hidden**

- Prvky, které se výhledově zobrazí (skriptování)

Stylování tabulek

- Na tabulkách lze stylovat
 - Celou tabulku - **<table>**
 - Řádky - **<tr>**
 - Buňky - **<td>, <th>**
 - Sloupce - **<col>, <colgroup>**
 - Záhlaví, zápatí, tělo - **<thead>, <tfoot>, <tbody>**
 - Nadpis - **<caption>**

Selektory

- Obvykle kontextové selektory
 - **table { ... }**
 - **table td { ... }**
 - **table.vysledky td { ... }**
 - **table th.hlavni { ... }**
 - **table tr#row1 { ... }**
- Využití kaskády:
 - **tr { ... }**
 - **table.vysledky tr { ... }**

Nadpisy tabulek <caption>

- Pozice nadpisu
 - **caption-side: top**
 - **caption-side: bottom**
 - Mozilla neoficiálně i **left** a **right**
- Ostatní vlastnosti
 - **text-align**
 - **vertical-align**
 - **width**

Nadpis tabulky - příklad

Tab 1. Nadpis nahoře

Jméno	Příjmení
Jan	Novák
Josef	Dvořák

Tab 2. Nadpis dole

Jméno	Příjmení
Jan	Novák
Josef	Dvořák

Rámečky v tabulkách

- Rámečky lze specifikovat odděleně pro buňky a celou tabulku
- Speciální vlastnosti:
 - **border-collapse: separate | collapse** - slučování okrajů buňek
 - **border-spacing: 5px** - vzdálenost okrajů buňek
 - **empty-cells: show | hide** - zobrazovat nebo nezobrazovat prázdné buňky

Příklady

```
table { border-collapse: separate }
```

Jméno	Příjmení
Jan	Novák
Josef	Dvořák

```
table { border-collapse: collapse }
```

Jméno	Příjmení
Jan	Novák
Josef	Dvořák

Příklady

```
table tr { border-top: 1px solid }
```

Jméno	Příjmení
Jan	Novák
Josef	Dvořák

```
table tr { border: 1px solid }
```

Jméno	Příjmení
Jan	Novák
Josef	Dvořák

Příklady

```
table { border: 1px solid }  
table th { border: 1px solid }  
table td { border: none;  
           border-left: 1px solid; }
```

Jméno	Příjmení
Jan	Novák
Josef	Dvořák

Pozadí tabulek a buněk

- Každá vrstva je buď průhledná (**border-color: transparent**) nebo má nastavenou barvu nebo obrázek
- Vrstvy se na sebe kladou v pořadí
 1. Tabulka **<table>**
 2. Skupiny sloupců **<colgroup>**
 3. Sloupce **<col>**
 4. Skupiny řádků **<thead>**, **<tbody>**, **<tfoot>**
 5. Řádky **<tr>**
 6. Buňky **<td>**, **<th>**

Stylování seznamů

- Stylují se položky seznamu (číslovaného či nečíslovaného)
- Můžeme definovat
 - **Typ odrážky** (nečíslované)
 - **Způsob čísluvání** (číslované)
 - **Obrázek místo odrážky**
 - **Pozice odrážky** (uvnitř textu nebo vně)

Typ odrážky

- Vlastnost **list-style-type**
- Aplikace na element ****

```
ul li { list-style-type: square; }
```

- Nečíslované seznamy:
 - **list-style-type: disc**
 - **list-style-type: circle**
 - **list-style-type: square**

Typ odrážky

- Vlastnost **list-style-type**
- Číslované seznamy:
 - i. **list-style-type: decimal**
 - ii. **list-style-type: lower-roman**
 - III. **list-style-type: upper-roman**
 - d. **list-style-type: lower-alpha**
 - E. **list-style-type: upper-alpha****list-style-type: none**

Obrázky v odrážkách

- Vlastnost **list-style-image**

```
<ul>
<li style="list-style-image: url('ok.gif')">
Položka seznamu</li>
</ul>
```

✓ Položka seznamu

Pozice odrážek

- Vlastnost **list-style-position**
- **list-style position: outside**
 - První položka v delším textu v delším textu v delším textu v delším textu
 - Druhá položka v delším textu v delším textu v delším textu v delším textu
- **list-style position: inside**
 - První položka v delším textu v delším textu v delším textu v delším textu
 - Druhá položka v delším textu v delším textu v delším textu v delším textu

Vlastnost list-style

- Shrnuje vlastnosti **list-style-***
- Formát:
 - **list-style: typ pozice obrázek**
- Příklady:
 - **list-style: square inside none**
 - **list-style: lower-alpha outside url("obr.gif")**

Kurzor myši

- Vlastnost **cursor** umožňuje zadat vzhled kurzoru myši umístěného nad příslušný prvek v dokumentu

Kurzor nad tímto

`slovem`
má jiný tvar.

Kurzor nad tímto slovem má jiný tvar.

Typy kurzoru

auto	crosshair	default
hand	pointer	move
text	wait	help

nw-resize	n-resize	ne-resize
w-resize		e-resize
sw-resize	s-resize	se-resize

- Ukazovací kurzor spolehlivě:
 - **cursor: pointer;** **cursor: hand**

Pseudoprvky

- Abstraktní označení některých prvků dokumentu nebo jejich stavu
- Definují vzhled částí dokumentu
 - **Pseudotřídy:** doplňkové charakteristiky elementů
 - Např. linky navštívené, nenavštívené
 - Specifičnost jako pro třídy
 - **Pseudoelementy:** části elementů nad rámec značkování
 - První řádek, první písmeno, ...
 - Specifičnost jako pro jména elementů
- Zápis pomocí :
 - **p:first-letter { font-size: 20px; }**

Pseudotřídy: stylování odkazů

- Statické
 - Nenavštívený odkaz:
a:link { color: green; }
 - Navštívený odkaz:
a:visited { color: red; }
- Dynamické
 - Přejetí myší:
a:hover { ... }
 - Aktivace odkazu:
a:active { ... }
 - Zaměření klávesnice:
a:focus { ... }

Typický styl odkazů

- Barevně zvýrazněný, po přejetí myší se podtrhne
- Při aktivaci zvýrazněný, po navštívení ztmavne

```
a { color: #aaf; text-decoration: none; }
a:link { }
a:visited { color: #339; }
a:hover { text-decoration: underline; }
a:active { text-decoration: underline; color: red; }
```

Příklad: [odkaz](#).

Úskalí pseudotříd

- Element `<a>` patří vždy do nějaké pseudotřídy
 - Ta se mění podle okolností
- Pravidlo `a { ... }` má menší prioritu než pravidla s pseudotřídou
 - S pseudotřídou se zachází stejně jako s třídou
- V tomto pravidle tedy nelze předefinovat nic, co už je definované v pseudotřídě

Pseudotřída :first-child

- Element, který je prvním potomkem svého otce

```
<ul class="menu">
<li>Modra polozka</li>
<li>Zelena polozka</li>
<li>Zelena polozka</li>
</div>
```

```
.menu li { color: green; }
.menu li:first-child { color: blue; }
```

- Obdobně **last-child**, v CSS3 mnohé další

Pseudoelementy: první písmena a řádky

- Element **:first-line**
 - Označuje první řádek libovolného elementu
 - Např. **p:first-line** - první řádek odstavce
 - Nelze předem určit, záleží na šířce okna, velikosti písma, ...
- Element **:first-letter**
 - Označuje první písmeno libovolného elementu
 - Např. **p:first-letter** - první písmeno odstavce

Příklad 1

```
p { text-transform: none; }  
p:first-line { text-transform: uppercase; }
```

<p>První řádek tohoto textu bude velkými
písmeny, ostatní řádky normálně. Další text následuje,
další text následuje a následuje a následuje.</p>

PRVNÍ ŘÁDEK TOHOTO TEXTU BUDE VELKÝMI PÍSMENY, OSTATNÍ ŘÁDKY
normálně. Další text následuje, další text následuje a následuje a
následuje.

Příklad 2

```
p:first-letter {  
    font-size: 200%;  
    font-weight: bold;  
    float: left;  
}
```

```
<p>První písmeno tohoto textu bude velké a obtékané  
zbývajícím textem. Ostatní písmena normálně.</p>
```

První písmeno tohoto textu bude velké a obtékané
zbývajícím textem. Ostatní písmena normálně.

Generovaný obsah

- Umožňuje generovat text, který v dokumentu není
- Pseudo-elementy **:before** a **:after**
 - **:before** – přidá text před obsah elementu
 - **:after** – za text elementu
- Vlastnost **content**:
 - Vlastní text, který bude vložen
- Všechny ostatní vlastnosti pro styl obsahu
 - Včetně **float**, **position**, **background** apod.

Příklad

```
div.pozn:before {  
    content: "Poznámka: ";  
    font-weight: bold;  
}
```

```
<div class="pozn">Toto pravidlo neplatí vždy</div>
```

Poznámka: Toto pravidlo neplatí vždy

6. Rozložení stránky

Praktické použití CSS

Layout

- Layout = rozložení, vizuální organizace jedné stránky
- Členění stránky na logické celky a jejich rozmístění
- Cíle
 - Přehlednost
 - Snadnost použití
 - Vizuální přitažlivost
- Prostředky
 - Rámy v HTML (extrémně zastaralé, mnoho nevýhod)
 - Tabulky v HTML (také zastaralé)
 - CSS

Tabulkový layout

- Zneužití tabulek pro vytvoření layoutu
- Příklad, skutečnost
- Kdysi jediný způsob => značně rozšířené
- Výhody
 - Jednoduchost návrhu
 - Kompatibilita
 - Mnoho možností manipulace s obsahem, zarovnávání, ...
- Nevýhody
 - Vzhled stránky daný přímo v HTML kódu
 - Dlouhý a nepřehledný kód (`<tr><td><tr><td>`)
 - Tabulka se zobrazí, až se celá načte.

CSS Layout

- Vizuální bloky jsou tvořeny nejčastěji pomocí HTML elementů pro tvorbu sekcí
 - Sémantické elementy `<header>`, `<footer>`, `<article>`, ...
 - Element `<div>` pro ostatní
 - Atributy `id` a `class`
- Rozmístění těchto bloků je definováno pomocí CSS
 - Šířka a výška
 - Okraje
 - Obtékání
 - Pozicování (absolutní či relativní)

Šířka stránky

- Plná šířka
 - Standardní stav (minimální okraje)
- Zúžená stránka s okrajem
 - Šířka závislá na velikosti okna prohlížeče
 - Zadaná relativně (v procentech)
- Stránka s pevnou šířkou
 - Šířka zadaná v absolutních jednotkách

Aspekty návrhu

- Jsme omezeni velikostí zobrazovací plochy
- Velký rozsah klientských zařízení
 - Desktop i 1600 px, mobil 320 px
- Dlouhé řádky se špatně čtou
 - Zúžit řádek
 - Zvětšit písmo
 - Více sloupců

Zúžená centrovaná stránka

```
<body>
    <div id="page">
        ... obsah stránky ....
    </div>
</body>
```

Zůžená centrováná stránka

```
body {  
    margin: 0; padding: 1em;  
}  
  
#page {  
    width: 80%;  
    margin: auto;  
    padding: 1em;  
}
```

Příklad

Pevná šířka

```
body {  
    margin: 0; padding: 1em;  
}  
  
#page {  
    width: 1000px;  
    margin: auto;  
    padding: 1em;  
}
```

Příklad

Umístování prvků vedle sebe

- V malém rozsahu pomocí tabulky
 - Pevné menší části stránky, prvky formuláře, ...
 - Triviální, dobře nastavitelné
- Absolutní pozicování
- Obtékání
- Speciální hodnoty **display:**

Absolutní pozicování

- Levý prvek absolutně pozicovaný vlevo
- Pravý prvek má levý okraj (**margin**)
- Příklad

```
#levy {  
    width: 5em;  
    position: absolute;  
    left: 0; top: 0;  
}  
#pravy {  
    margin-left: 5em;  
}
```

Absolutní pozicování

- Nevýhody
 - Určování vertikální absolutní pozice (uzavřít do DIVu)
 - Absolutně pozicovaný blok může překrývat předchozí i následující text

Absolutní pozicování (II)

- Absolutní pozicování k nadřazenému prvku
 - Nadřazený prvek musí mít nastaveno ***position: relative***
 - Příklad

```
#content {  
    width: 80%; margin: auto;  
    position: relative;  
}  
#menu {  
    width: 140px;  
    position: absolute;  
    right: -70px; top: 30px;  
}
```

Obtíkání

- Levý prvek je plovoucí vlevo
- Pravý prvek obtéká
- Příklad

```
#levy {  
    width: 5em;  
    float: left;  
}  
#pravy {  
    width: 10em;  
}
```

- Potíže při rozdílných výškách: příklad

Obtíkání (II)

- Řešení: levý okraj
- Příklad

```
#levy {  
    width: 5em;  
    float: left;  
}  
  
#pravy {  
    width: 10em;  
    margin-left: 5em;  
}
```

Jiná varianta obtékání

- Problém: v obsahu nelze využívat vlastnost **clear**:
- Řešení: oba bloky plovoucí vlevo
 - Umístituji se vedle sebe na levou stranu
 - Libovolný počet bloků vedle sebe

```
#levy {  
    width: 5em;  
    float: left;  
}  
#pravy {  
    width: 10em;  
    float: left;  
}
```

Třísloupcový layout

- Tradiční rozložení stránek
- Obvykle sloupce menu – hlavní obsah – doplňkové informace
- Dříve řešen tabulkou (triviální)
 - Všechny nevýhody tabulky – zejména pomalost
- Pomocí CSS poněkud komplikované

Třísloupcový layout (II)

- Princip:
 - Levý sloupec: plovoucí vlevo, pevná šířka i výška.
 - Pravý sloupec: obdobně, plovoucí vpravo.
 - Obsah: případné okraje
 - Zápatí: **clear: both**
- Varianty:
 - Často všechny sloupce plovoucí vlevo
 - Obsahující „kontejner“ s **overflow: hidden**
- Výška sloupců se nepřizpůsobí délce obsahu
 - Řešit lze např. barevným pozadím
 - Příklad (Petr Staníček)

Speciální hodnoty display

- Inline block **display: inline-block**
 - Bloky rozmístované jako řádkové elementy
- Prvky tabulek **display: table-cell**
 - Vytvoří ***anonymní tabulku***
 - Obdobně **table-row**, **table**, **inline-table**
- Příklady

Flexbox

- Nástroj pro rozmístování obsahu v lokální části stránky
 - Formuláře, položky seznamů, ...
- Rodičovský element slouží jako **flex kontejner**
 - Nastavením **display: flex**
 - Definuje se **hlavní osa** a **vedlejší osa**
- Synovské elementy se stávají **flex položkami**

Vlastnosti kontejneru

- **flex-direction** – směr hlavní osy
- **flex-wrap** – zalamování na více řádků/sloupců
- **justify-content** – rozmístění v řádku/sloupci
- **align-items** – zarovnání položek

Vlastnosti položek

- **flex**: <flex-grow> <flex-shrink> <flex-basis>
- **flex-grow** – jak položka relativně roste, pokud je místo navíc
 - **flex-grow**: 0 – vůbec neroste
 - **flex-grow**: 2 – váha 2, podle součtu vah ostatních
- **flex-shrink** – jak se položka relativně zmenší, pokud chybí místo
 - **flex-shrink**: 1
- **flex-basis** – základní velikost
 - **flex-basis**: auto
 - **flex-basis**: content
 - **flex-basis**: 10em
 - **flex-basis**: 40%

Grid layout

- Rodičovský element slouží jako **grid kontejner**
 - Nastavením **display: grid**
 - Nastaví se počet řádků a sloupců mřížky
 - Explicitní (template) nebo implicitní (auto)
 - Délková jednotka **fr** (fraction)
 - Možnost vytvoření pojmenovaných oblastí (více buněk mřížky)
- Pro synovské elementy se nastaví poloha v mřížce
 - **grid-row-start**, **grid-row-end** (**grid-row**)
 - **grid-column-start**, **grid-column-end** (**grid-column**)
 - nebo **grid-area** ([demo](#))
- [Popis a příklad na MDN](#)
- [Podrobnější popis](#)

Ukázky typických řešení

Odkazy jako tlačítka

```
<a href="..."><em>E-mail</em> burgetr@fit.vutbr.cz</a>
```

E-mail burgetr@fit.vutbr.cz

Odkazy jako tlačítka

```
<a class="button" href="...>
    <em>E-mail</em><span> burgetr@fit.vutbr.cz</span></a>

.button:link {
    margin: 0 0.2em; padding: 0.1em 0;
    border: 2px solid black;
    text-decoration: none;
    background: #ccc;
    color: black;
}
```

E-mail burgetr@fit.vutbr.cz

Odkazy jako tlačítka

```
.button em {  
    font-style: normal;  
    margin: 0; padding: 0.1em 0.5em;  
    background: white;  
    color: black;  
}
```

E-mail burgetr@fit.vutbr.cz

Odkazy jako tlačítka

```
.button span {  
    margin:0; padding: 0.1em 0.5em 0.1em 0.3em;  
}
```

E-mail burgetr@fit.vutbr.cz

Odkazy jako tlačítka

```
.button:hover {  
    background: #666;  
    color: white;  
}  
  
.button:hover em {  
    background: black;  
    color: white;  
}
```

E-mail burgetr@fit.vutbr.cz

Odkazy s ikonou

```
<a href="...>Nějaký odkaz</a>
```

[Nějaký odkaz](#)

Odkazy s ikonou

```
a {  
    background: #eeeeff url('newwin32.png')  
        top right no-repeat;  
    padding-right: 20px;  
    padding-top: 15px;  
}
```

[Nějaký odkaz](#)

Menu pomocí seznamů (Listamatic)

```
<ul id="menu">
    <li><a href="#">Položka 1</a></li>
    <li><a href="#">Položka 2</a></li>
    <li><a href="#">Položka 3</a></li>
</ul>
```

- Položka 1
- Položka 2
- Položka 3

Menu pomocí seznamů

```
ul#menu {  
    padding: 0 1px 1px;  
    margin-left: 0;  
    background: gray;  
    width: 13em;  
}
```

- Položka 1
- Položka 2
- Položka 3

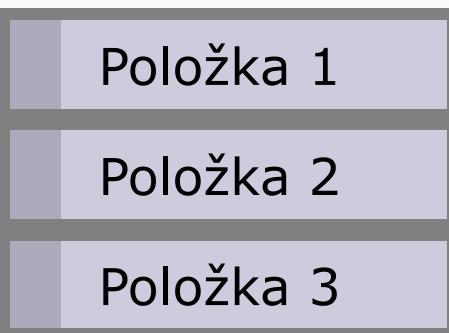
Menu pomocí seznamů

```
ul#menu li {  
    list-style: none;  
    margin: 0;  
    border-top: 1px solid gray;  
    text-align: left;  
}
```

Položka 1
Položka 2
Položka 3

Menu pomocí seznamů

```
ul#menu li a {  
    display: block;  
    padding: 0.25em 0.5em 0.25em 0.75em;  
    border-left: 1em solid #AAB;  
    background: #CCD;  
    text-decoration: none;  
}
```



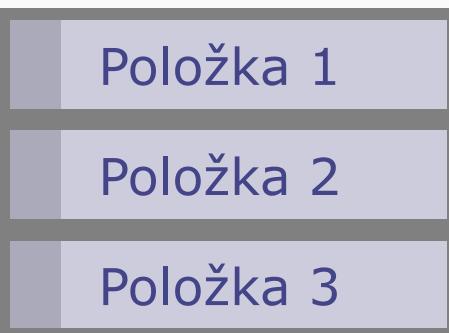
Položka 1

Položka 2

Položka 3

Menu pomocí seznamů

```
ul#menu li a:link { color: #448; }
ul#menu li a:visited { color: #667; }
ul#menu li a:hover {
    border-color: #FE3;
    color: #FFF;
    background: #332;
}
```



Položka 1

Položka 2

Položka 3

Plovoucí nabídky

```
ul#menu {  
    padding: 0 1px 1px;  
    margin-left: 0;  
    background: gray;  
    width: 13em;  
    float: left;  
    margin-right: 1em;  
}
```

Příklad

Plovoucí nabídky

```
body {  
    border: 2px solid gray;  
    margin-left: 5em;  
}
```

Příklad

Plovoucí nabídky

```
ul#menu {  
    padding: 0 1px 1px;  
    margin-left: 0;  
    background: gray;  
    width: 9em;  
    float: left;  
    margin-right: 1em;  
    position: relative;  
    left: -4em;  
}
```

Příklad

Plovoucí nabídky

```
ul#menu {  
    padding: 0 1px 1px;  
    margin-left: 0;  
    background: gray;  
    width: 9em;  
    float: left;  
    margin-right: -2em;  
    position: relative;  
    left: -4em;  
}
```

Příklad

Vodorovný seznam

```
<ul>
  <li><a href="#">Položka 1</a></li>
  <li><a href="#">Položka 2</a></li>
  <li><a href="#">Položka 3</a></li>
</ul>
```

- Položka 1
- Položka 2
- Položka 3

Vodorovný seznam

```
ul.vertlist li { display: inline; }

<ul class="vertlist">
  <li><a href="#">Položka 1</a></li>
  <li><a href="#">Položka 2</a></li>
  <li><a href="#">Položka 3</a></li>
</ul>
```

Položka 1 Položka 2 Položka 3

Vodorovný seznam

```
ul.vertlist li { display: inline-block; }

<ul class="vertlist">
  <li><a href="#">Položka 1</a></li>
  <li><a href="#">Položka 2</a></li>
  <li><a href="#">Položka 3</a></li>
</ul>
```

Položka 1 Položka 2 Položka 3

Vertikální centrování řádku

- Máme blok u kterého **známe výšku**
- Uvnitř máme **jednořádkový** text, který chceme vycentrovat

```
<div class="box">  
    Nějaký text  
</div>
```

Vertikální centrování řádku

```
#box {  
    height: 5em;  
    background-color: gray;  
}
```

Nějaký text

Vertikální centrování řádku

```
#box {  
    height: 5em;  
    line-height: 5em;  
    background-color: gray;  
}
```

Nějaký text

České odstavce

```
<style type="text/css">  
p { }  
</style>
```

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Sed elit magna, convallis interdum. Vestibulum rhoncus. Integer nibh pede, porttitor vulputate, cursus euismod, aliquam hendrerit, purus.

Integer nibh pede, porttitor vulputate, cursus euismod, aliquam hendrerit, purus. Vestibulum rhoncus.

České odstavce

```
<style type="text/css">  
p { text-indent: 30px; margin: 0px; }  
</style>
```

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Sed elit magna, convallis interdum. Vestibulum rhoncus. Integer nibh pede, porttitor vulputate, cursus euismod, aliquam hendrerit, purus.

Integer nibh pede, porttitor vulputate, cursus euismod, aliquam hendrerit, purus. Vestibulum rhoncus.

Pozadí stránky s barevným přechodem

- Vytváří pocit nekonečného pozadí
- Kombinace barvy pozadí a obrázku o šířce 1px
- Výhodou je minimální velikost obrázku
- Příklad

```
body {  
    background: url("pozadi.png") top repeat-x #c6c6c6;  
}
```

Image sprites

- Kompilace více malých obrázků do jednoho většího
- Jednotlivé obrázky jsou získány díky pozicování
- Rychlejší stahování - pouze jeden dotaz na obrázek
- Odstraňuje syndrom bliknutí v případě hover událostí
- Nutné zvolit vhodné rozmístění obrázků ve spritu - problém zobrazení
- Image sprite 1 - kombinování směrů
- Image sprite 2 - vertikální umístění

Image sprites (II)

- Praktický příklad 1

```
a.smile { background: url("imageSprite.png") -315px -28px; }  
a.smile:hover { background: url("imageSprite.png") -341px -28px; }  
  
a.del { background: url("imageSprite.png") -269px -52px; }  
a.del:hover { background: url("imageSprite.png") -269px -68px; }
```

- Praktický příklad 2

Fixní pozice hlavičky na stránce

- Při pobytu s obsahem je stále viditelná
- Fixní pozicování vzhledem k oknu stránky
- Příklad

```
#header {  
    position: fixed;  
    top: 0; left: 0;  
    width: 100%;  
    z-index: 1;  
}
```

Záložkové menu

- Kombinace bloku obsahu a seznamu (existuje několik řešení)
- Seznam vizuálně působí jako záložky
- Vytvoří se jeden styl pro aktivní založku a pro :hover nad založkou
- Pozadí obsahu a aktivní založka (:hover) musí být shodné
- Příklad

Centrovaný nadpis na lince

- Blok textu má horní rámček
- Součástí bloku je nadpis, který je pozicován nad linku
- Musí být nastaveno pozadí nadpisu, aby nerušilo orámování bloku
- Příklad

Jak používat CSS

Externí definice stylů

- Používejte externí definice stylů
 - Používejte nejlépe společnou definici pro všechny dokumenty
- Lze použít více stylových definic

```
<style type="text/css">
    @import "obecne.css";
    @import "mojesekce.css";
</style>
```

Prvky span a div

- <div> a se hodí pouze v případě, když nelze použít konkrétnější prvek
- Stylovat lze všechny prvky HTML
 - Zbytečná konstrukce
`<h1>Nadpis</h1>`
 - Lze zapsat rovnou
`<h1 style="...>Nadpis</h1>`

Atributy id a class

- Rozlišujte **class** a **id**
- Atribut **id**
 - Pro jedinečné prvky
 - Jednodušší stylování (specifický selektor)
 - Přehlednější styl
 - Kontrolovatelnost jedinečnosti
- Atribut **class**
 - Pro vlastnosti, které může mít více prvků

Názvy tříd a identifikátorů

- Volte vhodné názvy tříd a identifikátorů
 - Měly by vystihovat **význam**, nikoliv vzhled
 - Chybně: **pravysloupec**, **cervenepismo**
 - Správně: **menu**, **dulezite**
- Používejte stejné názvy tříd
 - Mnoho názvů se špatně pamatuje
 - Možno využít kontextové selektory

```
p span.prvni {...} /* pro první písmeno odstavce */
p.prvni {...} /* pro první odstavec v článku */
li.prvni {...} /* pro první položku seznamu */
```

Kontextové selektory

- Minimalizujte počet používaných tříd
- Chybně:

```
<h2 class="nadpisclanku">Nadpis</h2>
<p class="odstavecclanku">...</p>
...
<p class="odstavecclanku">...</p>
```

```
.nadpisclanku { ... }
.odstavecclanku { ... }
```

Kontextové selektory

- Správně:

```
<div class="clanek">
  <h2>Nadpis</h2>
  <p>...</p>
  ...
  <p>...</p>
</div>
```

```
.clanek h2 { ... }
.clanek p { ... }
```

Kontextové selektory

- Chybně:

```
<ul>
<li class="menu">...</li>
<li class="menu">...</li>
...
<li class="menu">...</li>
</ul>
```

```
.menu { ... }
```

Kontextové selektory

- Správně:

```
<ul class="menu">
<li>...</li>
<li>...</li>
...
<li>...</li>
</ul>
```

```
ul.menu li { ... }
```

Užitečné tipy (IV)

- Používejte více tříd pro jeden prvek

```
<div class="dulezitazprava">  
...  
</div>
```

- lépe:

```
<div class="zprava dulezite">  
...  
</div>
```

Využití kaskády

- Kaskádujte pravidla

```
h1, h2 {  
    font-family: "Arial CE", Arial, sans-serif;  
    font-weight: bold;  
}  
h1 {  
    font-size: 200%;  
}  
h2 {  
    font-size: 160%;  
}
```

Zápis pravidel

- Pište CSS styly přehledně
- Stanovte si pořadí selektorů
 - Elementy, id, třídy, ...
 - Nebo podle organizace stránky
- Stanovte si pořadí zápisu vlastností
 - Např. rozměry, okraje, pozice/obtékání, barvy a rámečky
- Pište pravidla na samostatné řádky
 - Snáze se pak kopírují
- Používejte komentáře /* **komentar** */

7. Pokročilé vlastnosti HTML, XML a XHTML

Záhlaví a zápatí tabulky

- Rozšiřující značky
 - **<thead>** - záhlaví tabulky
 - **<tfoot>** - zápatí tabulky
 - **<tbody>** - tělo tabulky
- Zadáváme vždy v tomto pořadí
- Uvedené značky obsahují **řádky** tabulky (každá část alespoň jeden řádek)
- Značka **<tbody>** se může vyskytnout opakovaně, je povinná, pokud tabulka neobsahuje pouze jednu část
- Spíše logický význam, vizuálně se (zatím) neprojeví

Příklad tabulky

```
<table>
  <thead>
    <tr><th>Jméno</th><th>Příjmení</th></tr>
  </thead>
  <tfoot>
    <tr>
      <td colspan="2">Strana 1</td>
    </tr>
  </tfoot>
  <tbody>
    <tr><td>Jan</td><td>Novák</td></tr>
    <tr><td>Josef</td><td>Dvořák</td></tr>
  </tbody>
</table>
```

Příklad tabulky

Jméno	Příjmení
Jan	Novák
Josef	Dvořák
Strana 1	

Formátování tabulky po sloupcích

- Někdy je účelné specifikovat zarovnávání a další atributy pro jednotlivé sloupce nebo jejich skupiny
- V HTML lze použít značky
 - **<col>** – jeden nebo více sloupců se společnými atributy
 - **<colgroup>** - skupina sloupců (strukturální)
- Definice sloupců ovlivňují pouze strukturu, nikoliv obsah
- Definice sloupců není povinná - potom se celá tabulka považuje za jedinou skupinu sloupců

Definice sloupců

```
<table>
  <col width="10" align="center" valign="bottom">
  <col span="3" width="15">
  <tr><td>...
  ...
</table>
```

- Počet sloupců musí odpovídat skutečnému počtu buněk na řádcích

Skupiny sloupců

```
<table>
  <colgroup span="5" width="10"></colgroup>
  <colgroup width="15">
    <col align="center" valign="bottom">
    <col span="3" align="right">
  </colgroup>
  <tr><td>...
  ...
</table>
```

- Pokud skupina obsahuje alespoň jeden `<col>`, hodnota jejího atributu **span** se ignoruje.

Link relations

- Doplňuje sémantiku používaných odkazů – popisuje důvod odkazování na jiný dokument
- Rozšíření množiny hodnot od verze v HTML4
- Použití u elementů ***link, a, area***
- Dvě kategorie linků:
 - odkaz na externí zdroj – připojení obsahu k aktuálnímu dokumentu

```
<link rel="stylesheet" href="style.css" type="text/css">
```

- hyperlink – link vedoucí na jiný dokument

```
<a href="slide010.html" rel="next">next</a>
```

Link relations (II)

- Nové hodnoty atributu ***rel***:
 - **author** – odkaz na autora
 - **archives** – odkaz na starší obsah
 - **external** – odkaz na externí stránku
 - **licence** – odkaz na licenci o sdílení obsahu
 - **tag** – odkaz na stránku jež je kategorií obsahu (například u blogů)
 - **search** – odkaz na stránku poskytující vyhledávání
 - **first, last** – odkaz na první/poslední dokument
 - **prev, next** – odkaz na předcházející/následující dokument
 - **up, down** – odkaz na nadřazený/podřazený dokumentu
 - atd.

Mikrodata

- Standardizovaný způsob zavedení sémantiky elementům
- Zjednodušení strojového zpracování
- Atributy:
 - **itemscope** – kontejner pro sérii položek
 - **itemprop** – konkrétní vlastnost
 - **itemtype** – typ element reprezentován pomocí URL
 - **itemid** – globální identifikátor (např. ISBN knihy)
 - **itemref** – odkaz na konkrétní identifikátor

Mikrodata (II)

- Příklad

```
<div itemscope itemtype="http://example.org/person">
  <p>Moje jméno je <span itemprop='name'>Josef</span>.
  </p>
  <p>Moje příjmení je <span itemprop='surname'>Novák</span>.
  </p>
  <p>Byl jsem narozen v <span itemprop='born-
  city'>Brně</span>.</p>
</div>
```

Multimedia

- Nové multimedialní elementy:
 - **audio** – definice zvukového obsahu
 - **video** – definice videa nebo filmu
 - **source** – definice zdroje pro **video** nebo **audio**
 - **embed** – definice kontejneru pro externí aplikaci nebo interaktivní obsah
 - **track** – definice textu skladby

Multimedia <audio>

- Atribut **controls** zobrazí ovládací prvky (play, stop, pause)
- Přehrávání je možno ovládat pomocí JavaScriptového API

```
<audio controls>
  <source src="sound.ogg" type="audio/ogg">
  <source src="sound.mp3" type="audio/mpeg">
  Prohlížeč nepodporuje tag audio.
</audio>
```

- Podporované formáty:
 - MP3
 - Wav
 - Ogg

Multimedia <video>

```
<video width="320" height="240" controls>
    <source src="file.mp4" type="video/mp4">
    <source src="file.ogg" type="video/ogg">
    Prohlížeč nepodporuje tag video.
</video>
```

- Podporované formáty:
 - H.264 (MP4)
 - WebM
 - Ogg Theora (ogv)

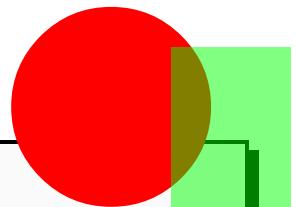
Multimedia <video> s Flash fallbackem

```
<video width="640" height="360" controls>
  <source src="v.mp4" type="video/mp4" />
  <source src="v.ogv" type="video/ogg" />
  <object width="640" height="360"
    type="application/x-shockwave-flash"
    data="player.swf">
    ...
  </object>
</video>
```

Grafika

- Vnořené SVG a MathML přímo v dokumentu

```
<svg height="120"  
      xmlns="http://www.w3.org/2000/svg">  
  <circle cx="50" cy="50" r="50" fill="red" />  
  <rect x="80" y="20" height="60" width="60"  
        style="stroke:none; fill:#00ff00;  
        fill-opacity: 0.5;" />  
</svg>
```



Prvek <canvas>

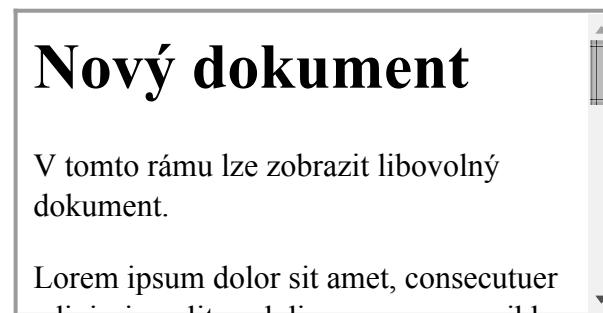
- Kreslicí plátno
- Kreslení pomocí JavaScript API

```
<canvas id="square">
    fallback content
</canvas>

<script>
    canvas = canvas.getElementById('square');
    context = canvas.getContext('2d');
    context.fillStyle = "#000000";
    context.fillRect(0, 0, 100, 100);
</script>
```

Vložený rám

```
<iframe src="frames_c5.html" width="150" height="100">  
Vložené rámce nejsou podporovány.  
</iframe>
```



V dnešní době je používáno k zobrazení fanoušků na Facebooku.

Dodatečné informace v hlavičce

- Meta informace - dodatečné informace o dokumentu
- Vždy v hlavičce dokumentu
- Dva možné tvary:
 - Uživatelské informace

```
<meta name="..." content="..."></meta>
```

- Ekvivalent HTTP hlavičky

```
<meta http-equiv="..." content="..."></meta>
```

Uživatelské informace

- Popis dokumentu

```
<meta name="description" content="Stručný popis">
```

- Klíčová slova

```
<meta name="keywords" content="WWW, HTML, stránky">
```

- Autor dokumentu

```
<meta name="author" content="Jan Novák">
```

Uživatelské informace (II)

- Generátor (software)

```
<meta name="generator" content="Mozilla Composer">
```

- Informace pro roboty

```
<meta name="robots" content="index, follow">
<meta name="robots" content="noindex,nofollow">
```

Vlastní meta tagy

- **name** a **content** můžou obsahovat téměř cokoliv aniž bychom narušili význam stránky
- Vhodné pro programy, které pracují se stránkami
- Např. specifické nastavení prohlížečů (tablety, mobilní telefony)

```
<meta name="viewport" content="width=device-width">
<meta name="viewport" content="user-
scalable=no, initial-scale=1">
```

Facebook meta tags

- Meta tagy využívané protokolem Open Graph od společnosti Facebook.
- Pomáhá načtení vhodných dat při sdílení stránek na Facebooku

```
<meta property="og:title" content="Název stránky">
<meta property="og:description" content="Popis stránky">
<meta property="og:site_name" content="Název-webu.cz">
<meta property="og:url" content="http://nazev-
webu.cz/stranka/">
<meta property="og:image" content="http://nazev-
webu.cz/obrazek.jpg">
<meta property="og:type" content="article">
```

HTTP ekvivalenty

- Stejná informace, jako v hlavičkách HTTP
- Není nutný přístup k nastavení serveru
- Typ obsahu = MIME typ + kódování

```
<meta http-equiv="content-type"  
      content="text/html; charset=windows-1250">
```

- Jazyk obsahu

```
<meta http-equiv="content-language"  
      content="cs">
```

Přesměrování

- Po zobrazení stránky se po stanovené době automaticky přejde na jiný dokument
- Např. po 6 vteřinách na Seznam.cz

```
<meta http-equiv="refresh"  
      content="6;URL=http://www.seznam.cz">
```

Ovládání cache

- Řídí zda ukládat nebo neukládat stránku v cache
- Ovlivňuje pouze cache v prohlížeči
- Např.

```
<meta http-equiv="cache-control"  
      content="cache">  
  
<meta http-equiv="cache-control"  
      content="no-cache">
```

Vypršení

- Nastavení doby, dokdy je dokument platný
- Poté může být vymazán z cache
- Např.

```
<meta http-equiv="expires"  
      content="Wed 14 Oct 2004 16:10:00">
```

Vazby na jiné dokumenty

- Značka **<link>** vytváří neviditelnou vazbu na jiný dokument.
- Vždy uveden v hlavičce dokumentu
- Běžné použití:
 - Ikony stránek
 - Externí definice CSS

```
<link rel="vztah" href="url" type="typ">
```

Ikona stránky

- Zobrazuje se v adresním řádku, v záložkách, ...
- Soubor s obrázkem ve formátu ICO uložený na serveru
- Propojení s HTML dokumentem (v hlavičce):

```
<link rel="shortcut icon"  
      href="http://muj.web.cz/favicon.ico"  
      type="image/x-icon">
```

- Většina prohlížečů hledá automaticky soubor **favicon.ico** v kořenovém adresáři i bez této specifikace

Podpora RSS a ATOM

- Rodina XML formátů podporující dodávání aktuálního obsahu
- Technologie umožňuje uživateli přihlásit se k odběru novinek přímo při prohlížení stránek

```
<link rel="alternate"
      type="application/atom+xml"
      title="Feed of recent questions"
      href="/feeds">

<link rel="alternate"
      type="application/rss+xml"
      title="Českénoviny.cz - Hlavní události"
      href="http://www.ceskenoviny.cz/sluzby/rss/index.php">
```

XML a XHTML

XML a XHTML

- XML
 - Obecný standard pro výměnu dat
 - Syntakticky podobný HTML, nedefinuje význam značek (záleží na konkrétní aplikaci)
- XHTML
 - Konkrétní aplikace XML pro webové stránky
 - Jména značek a sémantika jako v HTML

Výhody XML

- Jednodušší implementace
 - Jazyk je zjednodušen
 - Lépe se zpracovává
- Možnost ukládání, indexace a vyhledávání
 - Nativní XML databáze
- Mnoho nástrojů pro zpracování
 - Čtení dokumentů (DOM, SAX parsery)
 - Transformace (XSLT)
 - ...
- Mnoho aplikací je již založeno na XML

HTML a XHTML

- Společné vlastnosti
 - Stejný význam většiny značek a jejich atributů
 - Stejný způsob zápisu
- Rozdíly
 - Striktní syntaxe
 - Odstraněny značky a atributy pro formátování, nahrazeno CSS
 - Musí splňovat požadavky XML

Hlavička

- Každý XML dokument musí začínat hlavičkou

```
<?xml version="1.0" encoding="windows-1250"?>
```

- Pokud není uvedeno kódování, myslí se UTF-8 (Unicode)

Typ dokumentu (DTD)

- Přechodový

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0  
Transitional//EN"  
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-  
transitional.dtd">
```

- Striktní

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0  
Strict//EN"  
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
```

Typ dokumentu (DTD) (II)

- Rámce

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0
Frameset//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-frameset.dtd">
```

Kořenový element <html>

- Kořenový element <html> musí mít nastaveny atributy **xmlns** a **xml:lang**:

```
<html xmlns="http://www.w3.org/1999/xhtml"  
      xml:lang="cs" lang="cs">
```

- Musí obsahovat elementy <head> a <body>
- Element <head> musí obsahovat <title>

Vnořování značek

- Všechny značky musí být korektně vnořené
- Překrývání není povoleno ani v SGML (HTML), ale toleruje se
- XML striktně vyžaduje tzv. ***well-formed document***
- Překrývání elementů

```
<p>Toto je věta se <em>zdůrazněním.</p></em>
```

- Správné vnoření elementů:

```
<p>Toto je věta se <em>zdůrazněním</em>. </p>
```

Zápis atributů

- Hodnoty atributů **musí** být v úvozovkách
- Každý atribut musí mít hodnotu
 - HTML:
`<option name="a" checked>...</option>`
 - XHTML:
`<option name="a" checked="checked">...</option>`
- Hodnoty atributů, které jsou typu výčet jsou case-sensitive
 - `<form action="..." method="post">`
 - `<input type="submit">`
- Všechny znaky **&** v hodnotách atributů musí být zapsány pomocí **&**

Skripty a styly

- Skripty obsahující & a < musí být označeny jako CDATA:

```
<script type="text/javascript">
  <![CDATA[
    ... text skriptu ...
  ]]>
</script>
```

XHTML 1.1

- Specifikace XHTML 1.0 byla rozdělena na **moduly**
 - Základní moduly (XHTML Basic), formuláře, tabulky, ...
 - Umožňuje vytvořit jazyk „na míru“ aplikaci - tzv. **build**
- XHTML 1.1 je build navržený tak, aby odpovídal XHTML 1.0 Strict
- Změny oproti XHTML 1.0 Strict
 - Atribut **lang** odstraněn (zůstává **xml:lang**)
 - U elementů **<a>** a **<map>** byl atribut **name** nahrazen pomocí **id**

Dokument XHTML 1.1

- Kostra dokumentu:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.1//EN"
  "http://www.w3.org/TR/xhtml11/DTD/xhtml11.dtd">
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en">
  <head>
    <title>...</title>
  </head>
  <body>
    ...
  </body>
</html>
```

Struktura dokumentu

- Řádkové prvky nesmí obsahovat blokové (i v HTML)
- Prvky **body** a **form** nesmí přímo obsahovat
 - Běžný text
 - Většinu řádkových prvků včetně obrázků
- Řešení: uzavřít veškerý text do odstavců **<p>** nebo bloků **<div>**
- U formulářů to platí pro text i všechny prvky formuláře

XHTML a přenosový protokol

- XHTML by měl být v prohlížeči zpracován XML parserem
- To musí být indikováno MIME typem při HTTP přenosu
 - Typ **text/html** pro HTML dokumenty
 - Typ **application/xhtml+xml** pro XHTML dokumenty
- Praktické problémy
 - Nutnost konfigurace na straně serveru
 - MSIE neakceptuje **application/xhtml+xml** (snaží se ukládat na disk)
- Prozatímní běžné řešení
 - XHTML dokument + **text/html** MIME typ

Důsledky prozatímního řešení

- Prohlížeč zpracovává XML dokument HTML parserem
 - Specifikace XHTML 1.0 to připouští
 - Doporučuje dodatečná syntaktická omezení za účelem minimalizovat zmatení prohlížeče
- HTML parser je tolerantní k chybám
 - Zpracovává dokument jako tzv „tag soup“
 - Nekontroluje, zda je dokument well-formed XML
 - Nedokáže interpretovat pokročilejší možnosti XML
 - Namespaces, definice vlastních entit, ...
 - Ukázky
- XML prolog **<?xml version="1.0" encoding="utf-8"?>**
 - Může zmást detekci typu dokumentu (MSIE 6)
 - Může se vynechat, ale dokument musí být v UTF-8

Budoucnost XHTML

- XHTML 1.1 je poslední stabilní verze
- Rozpracované byly další
 - XHTML 1.2 (evoluce)
 - XHTML 2 (revoluce)
- Problémy XHTML 2
 - Modernizace a příklon k sémantickému webu
 - Opuštění zpětné kompatibility
 - Velmi pomalý postup
 - Negativní přijetí vývojáři prohlížečů (neochota implementovat) i velkou částí uživatelů

(X)HTML 5

- Iniciativa vzniklá nezávisle na W3C jako reakce na problémy vývoje XHTML2
 - Alternativní skupina WHATWG
 - Otevřenější vývoj, kdokoliv se může zapojit
 - V roce 2007 uznáno W3C
- Vychází z HTML 4, přidává nové možnosti
 - Nové značky pro sémantiku obsahu, audio, video a další
- Definuje názvy a význam značek a atributů **nezávisle na syntaxi**
- Syntaktický zápis je možno zvolit
 - HTML serializace
 - XML serializace (někdy nazývaná XHTML 5)

HTML5 Syntax

- HTML syntax
 - MIME typ **text/html**
 - Jsou definována pravidla pro parsování včetně zpracování chyb
- XHTML syntax
 - MIME typ **application/xhtml+xml** nebo **application/xml**
 - Parsování dáno pravidly XML

Hlavíčka dokumentu

- Nový univerzální typ dokumentu **html**

```
<!DOCTYPE html>
```

(pouze kvůli přepínání režimů)

- XHTML bez DOCTYPE, ale s definicí namespace

```
<?xml version="1.0" encoding="UTF-8"?>
<html xmlns="http://www.w3.org/1999/xhtml">
...
</html>
```

Obecné XML

- Prohlížeč může zpracovat obecné XML
- Není žádný výchozí styl
 - Je ale možné definovat vlastní
 - Direktiva **xmlstylesheet**
- Definice stylu
 - **CSS**
 - Definice vzhledu jednotlivých elementů
 - **XSLT**
 - Interní transformace na HTML dokument
 - Včetně JavaScriptu atd.
- Ukázky

Odkazy

- Rozdíly HTML 4 a 5 (oficiální)
<http://dev.w3.org/html5/html4-differences/>
- Test prohlížeče na HTML 5 + aktuální žebříčky
<http://html5test.com/>
- Aktuální použitelnost jednotlivých vlastností
<http://html5please.com/#html>
- Podrobnější vysvětlení významu elementů
<http://html5doctor.com/element-index/>
- <http://caniuse.com/> - zjištění kompatibility

8. Média a CSS 3

Import stylů

- Pravidlo **@import**
 - `@import url("mujstyl.css");`
 - `@import "mujstyl.css";`
- Musí předcházet všechna ostatní pravidla v CSS definici
 - Importované definice předchází všechny místní definice
 - Následující pravidla mohou předefinovat importované styly
 - Opačně nikoliv

Externí předpis znovu

- Tradiční způsob vložení externího předpisu

```
<!DOCTYPE ... >
<html>
<head>
    <title>Titulek</title>
    <link rel="stylesheet" type="text/css"
          href="definice.css">
</head>
<body>
    ...

```

Externí předpis znova (II)

- Vložení pomocí @import

```
<!DOCTYPE ... >
<html>
<head>
    <title>Titulek</title>
    <style type="text/css">
        @import "definice.css";
    </style>
</head>
<body>
    ...

```

Média

- CSS definuje různé typy **médií** na kterých je možno zobrazovat dokument
- Typy médií jsou sdruženy do skupin podle svých charakteristik
- Pravidla stylu mohou být přiřazena médiu
 - Dokument může vypadat jinak na obrazovce než na tiskárně
- Používaný typ média určuje klient (prohlížeč)
- V jednom okamžiku je platný vždy **právě jeden** typ média
- Některé CSS vlastnosti jsou použitelné pouze pro určitou skupinu médií

Typy médií v CSS

- **aural** - zvukový výstup (syntetizátor)
- **braille** - braillův řádek (dotykový)
- **embossed** - braillův tisk (plastický)
- **handheld** - obrazovka přenosného počítače
- **print** - tisk nebo náhled tisku
- **projection** - prezentace
- **screen** - běžná obrazovka
- **tty** - znakový terminál, dálnopis
- **tv** - televizní obrazovka

Skupiny médií

- Stránkovaná – plynulá
 - emboss, handheld, print, projection, tv
 - aural, braille, handheld, screen, tty, tv
- Zvuková – vizuální – dotyková
 - aural, tv
 - handheld, print, projection, screen, tty, tv
 - braile, emboss

Skupiny médií (II)

- S pevnou mřížkou – bitmapová
 - braille, emboss, handheld, tv
 - handheld, print, projection, tv
- Interaktivní – statická
 - aural, braille, handheld, projection, screen, tty, tv
 - aural, braille, emboss, handheld, print, projection, screen, tty, tv

Specifikace média

- Podmínka při importu sady stylů
 - `@import "obrazovka.css" screen;`
 - `@import "tisk.css" print, emboss;`
 - `@import "ostatni.css" all;`
- Při vazbě na HTML dokument

```
<link rel="stylesheet" type="text/css" media="print">

<style type="text/css" media="screen, handheld">
...
</style>
```

Specifikace média (II)

- Uvnitř definice pomocí pravidla @media

```
@media screen {  
    body {  
        font-family: sans-serif;  
        font-size: 12pt;  
    }  
}  
@media print {  
    body {  
        font-family: serif;  
        font-size: 10pt;  
    }  
}
```

Stránkovaná média

- Obvykle tisk nebo projekce
- Nastavujeme způsob stránkování před, za a uvnitř konkrétního prvku
- Stránkování se řídí vlastnostmi
 - **page-break-before: auto | always | avoid | left | right**
 - **page-break-after: auto | always | avoid | left | right**
 - **page-break-inside: auto | avoid**

Režimy stránkování

- **auto** - rozhodne prohlížeč
- **avoid** - zákaz zlomu stránky
- **always** - vynucení zlomu stránky
- **left** - přechod na další levou stránku (případně se vloží prázdná stránka)
- **right** - přechod na další pravou stránky
- Pozor: omezená podpora v prohlížečích

Příklad

- Styl prezentace

```
...
<div class="slide">
    Obsah snímku
</div>
...
```

```
div.slide {
    page-break-after: always;
    page-break-inside: avoid;
}
```

Media queries

- Rozšíření specifikace ve srovnání s CSS2
- Umožňují podmínit pravidla vlastnostmi zařízení (média)
- Aktivace kaskádových stylů podle vlastností zařízení
- Media queries jsou důležitou součástí **responsivního designu**
- Media query se skládá z
 - typ zařízení (media type)
 - vlastnost zařízení (media feature)
 - hodnota (value)

Media features

- **width/height** – šířka/výška zobrazované oblasti dokumentu
- **device-width/device-height** – šířka rendrované oblasti (rozlišení)
- **orientation** – orientace displeje
- **aspect-ratio** – poměr výšky a šířky
- **device-aspect-ratio** – poměr výšky šířky pro device hodnoty

Media features

- **color** – počet bitů na jednu barvu
- **color-index** – počet barev v indexové tabulce barev
- **monochrome** – podpora monochromatického zařízení
- **resolution** – rozlišení
- **scan** – způsob vykreslování obrazu (media type **tv**)
- **grid** – volba vykreslování po znacích nebo po pixelech (media type **tty**)

Media query – příklady

- Konkrétní zařízení (nevhodné):

```
@media screen and (device-width: 320px)  
    and (device-height: 480px) {  
    ...  
}
```

- Mobilní telefony:

```
@media handheld,  
    only screen and (max-device-width: 300px) {  
    ...  
}
```

Mobilní zařízení

Pro podporu mobilních zařízení (telefony, tablety):

1. Vypnout automatické škálování obrazu (v hlavičce HTML):

```
<meta name="viewport" content="width=device-width, initial-scale=1">
```

2. Přidání specifických stylů podle velikosti displeje (**breakpoints**):

```
<link rel="stylesheet" href="main.css" type="text/css">
<link rel="stylesheet" href="tablet.css" type="text/css"
      media="screen and (max-width: 1365px) and (min-
      width: 876px)">
<link rel="stylesheet" href="mobile.css" type="text/css"
      media="screen and (max-width: 875px)">
```

CSS Frameworks

- Předem připravené stylové předpisy (mohou zahrnovat i JavaScript), které řeší nejtypičtější úlohy:
 - Rozložení obsahu pomocí mřížky
 - Základní styl záhlaví, zápatí, nadpisů, seznamů, atd.
 - Obecné předdefinované CSS třídy: zvýrazňování, základní barvy, ...
- Nejznámější framework: Bootstrap
 - Příklad, oficiální příklady
- Další frameworky: Foundation, Susy, ...

CSS Frameworks: Plusy a mínusy

- Výhody:
 - Rychlé prototypování stránek
 - Předdefinovaný, dobré vypadající design
 - Responsivní již v základu
- Nevýhody:
 - Uniformní vzhled
 - Vzhled je vlastně definován v HTML (použitím různých tříd a pomocných elementů)
 - Nehezký kód, špatná udržovatelnost
 - Uživatelské změny jsou obtížnější

CSS Level 3

Úvod do CSS 3

- Stále v procesu návrhu (současný stav)
- Většina navrhovaných vlastností je již prohlížeči podporována
 - avšak je nutné testování
- Některé prohlížeče vyžadují předponu před názvem vlastnosti
 - **-moz-** Firefox
 - **-o-** Opera
 - **-ms-** IE
 - **-webkit-** Chrome, Safari
- Novinky
 - Rozšíření množiny selektorů
 - Nové vlastnosti prvků
 - Podpora media queries
 - a další

Nové selektory

- `[foo^="bar"]` – elementy, kde hodnota atributu začíná "bar"
- `[foo$="bar"]` – elementy, kde hodnota atributu končí "bar"
- `[foo*="bar"]` – elementy, kde hodnota atributu obsahuje "bar"
- `:enable` – element má definovanou vlastnost **enable**
- `:checked` – element je označen (např. **radio-button** nebo **checkbox**)
- `:not(s)` – element, který neodpovídá selektoru **s**
 - Např. `*:not(:link):not(:visited)`(vše kromě odkazů)
- `E ~ F` – element F je předcházen elementem E (sourozenci)
 - Srovnej s `E + F`

Nové selektory (II)

- **:root** – element je kořenem dokumentu
- **:nth-child(an+b)** – element je n-tý potomek rodičovského elementu
- **:nth-last-child(an+b)** – element je n-tý potomek rodičovského elementu (počítáno od konce)
- **:nth-of-type(an+b)** – element je n-tý v pořadí daného typu konkrétního rodiče
- **:nth-last-of-type(an+b)** – element je n-tý v pořadí daného typu (počítáno od konce)
- **:last-child** – poslední element rodičovského elementu

Příklad

```
ul.stripped li {  
    list-style-position: inside;  
    color: blue;  
}  
ul.stripped li:nth-child(2n+1) {  
    color: white;  
    background-color: navy;  
}
```

- První položka
- Druhá položka
- Třetí položka
- Čtvrtá položka
- Pátá položka

Nové selektory (II)

- **:first-of-type** – element je první daného typu pro konkrétního rodiče
- **:last-of-type** – element je poslední daného typu pro konkrétního rodiče
- **:only-child** – element je jediný potomek rodiče
- **:only-of-type** – elementy rodiče jsou pouze jednoho typu
- **:empty** – element nemá potomky (ani textové)

Kompletní přehled selektorů

Definice barvy s podporou průhlednosti

- Rozšíření definice barvy o alfa kanál (průhlednost)
- **RGBA (Red, Green, Blue, Alpha)**

```
a {  
    background: rgba(255, 0, 0, 0.5);  
    /* poloprůhledná červená */  
}
```

- **HSLA (Hue, Saturation, Lightness, Alpha)**

```
a {  
    background: hsla(240, 100%, 50%, 0.5);  
    /* poloprůhledná modrá */  
}
```

Pozadí – gradient

- Umožňuje definovat barevný přechod
- Typy gradientů:
 - **linear-gradient**
 - **radial-gradient**
 - **repeating-linear-gradient**
 - **repeating-radial-gradient**
- Příklad:

```
div {  
    background-image:  
        linear-gradient(45deg, white, #555);  
}
```

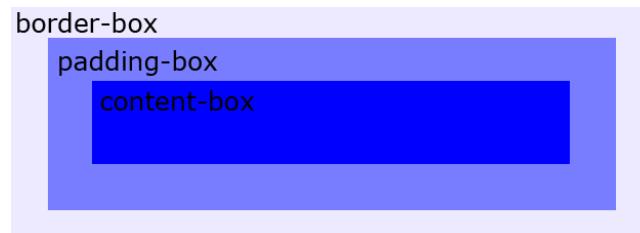
Pozadí – více obrázků na pozadí

- Jeden element může mít definován libovolný počet obrázků na pozadí
- Pro každý obrázek je možné nastavit specifické vlastnosti
- Příklad:

```
div {  
    background-image: url(1.jpg), url(2.jpg), url(3.jpg);  
    background-repeat: no-repeat, no-repeat, repeat;  
    background-position: right bottom, left bottom, 0 0;  
}
```

Pozadí – nové vlastnosti

- **background-clip** – ořezání obrázku v návaznosti na box model
- **background-origin** – definuje umístění obrázku v rámci box modelu
 - content-box
 - padding-box
 - border-box



- **background-size** – velikost obrázku na pozadí

Pozadí – nové vlastnosti (II)

- Příklad:

```
div {  
    background: url(image1.jpg);  
    background-size: 80px 60px;  
    background-repeat: no-repeat;  
    background-origin: content-box;  
}
```

Web fonts

- Umožňuje definovat uživatelské fonty
- Nový font se definuje pomocí pravidla **@font-face**
- @font-face má následující parametry:
 - **font-family** – pojmenování uživatelského fontu
 - **src** – cesta k souboru s fontem
 - **font-style** – normální font nebo kurzíva
 - **font-weight** – tučnost písma
 - **font-stretch** – řez písma

Web fonts (II)

- Příklad:

```
@font-face {  
    font-family: 'NovyFont';  
    font-weight: bold;  
    src: url('newFontFile.woff') format('woff'),  
        url('newFontFile.ttf') format('truetype'),  
        url('newFontFile.svg') format('svg');  
}  
  
h1 {  
    font-family: 'NovyFont', sans-serif;  
    font-weight: bold;  
}
```

Nové vlastnosti formátování textu

- **hanging-punctuation** – umístění interpukční značky
- **text-align-last** – zarovnání posledního řádku textu
- **text-emphasis** – zvýraznění toku textu východních jazyků
- **text-overflow** – definice chování pokud je text delší než element
- **text-shadow** – stínování textu
- **text-wrap** – definice zalamování řádku textu
- **word-break** – nastavení jak budou zalamována slova v elementu
- **word-wrap** – povolení dělení slov
- atd.

Nové vlastnosti formátování textu (II)

- Příklad:

```
h1 {  
    text-shadow: 5px 5px 5px #00FF00;  
}  
  
p {  
    width: 400px;  
    word-wrap: break-word;  
}
```

Multi-column

- Definice textu do více sloupců
- Je možné definovat počet sloupců, jejich rozměry, předěly atd.
- **column-width** – šířka sloupce
- **column-count** – počet sloupců
- **columns** – umožňuje najednou definovat **column-width** a **column-count**

```
div {  
    columns: auto 2;  
}
```

Multi-column (II)

- **column-gap** – velikost mezery mezi sloupci
- **column-rule** – definuje styl oddělení dvou sloupců (podobný jako **border**)
 - **column-rule-color** – barva
 - **column-rule-style** – styl
 - **column-rule-width** – tloušťka

```
div { column-rule: 1px dotted #fff; }
```

```
div {  
    column-rule-color: #fff;  
    column-rule-style: dotted;  
    column-rule-width: 1px;  
}
```

Multi-column (III)

- **column-span**
 - vnořený prvek je vyjmut ze sloupcového rozložení
 - např. nadpis přes všechny sloupce

```
h2 {  
    column-span: all;  
}
```

- **column-fill** – definuje styl rozložení textu do sloupců (má smysl pro tiskový styl)

Kulaté rámečky

- **border-radius** – kulatý rámeček
 - Pořadí rohů – levý horní, pravý horní, pravý spodní a levý spodní
 - Pokud jsou zadány dvě hodnoty
 - první hodnota zaoblení je pro levý horní a pravý dolní
 - druhá hodnota zaoblení je pro pravý horní a levý dolní
- Příklad – poloměr zaoblení na ose **y** a **x**:

```
div {  
    border-top-left-radius: 15pt 25pt;  
}
```

Rámeček s obrázkem

- **border-image: source slice width outset repeat;**
 - **border-image-source** – umístění obrázku k orámování
 - **border-image-slice** – definice 4 přímek rozdělujících zdrojový obrázek na 9 částí
 - **border-image-width** – definuje tloušťku rámečku
 - **border-image-outset** – posunutí vykreslování
 - **border-image-repeat** – definuje chování rozděleného obrázku (9 částí)
- Příklad – source slice repeat;

```
p {  
    border-  
    image: url(border.png) 27 27 27 27 stretch round;  
}
```

Stínování boxu – box-shadow

- Definice podbarvení elementu
- **box-shadow:** `inset x y poloměr tloušťka barva;`
 - **inset** – pokud je nastaveno, tak podbarvení směruje dovnitř (nepovinné)
 - **x** – vzdálenost posunu stínu po ose X
 - **y** – vzdálenost posunu stínu po ose Y
 - **poloměr** – poloměr rozmazání
 - **tloušťka** – poloměr tloušťky podbarvení
 - **barva** – barva podbarvení

```
div {  
    box-shadow: 10px 10px 10px 10px blue;  
}
```

2D transformace

- Transformace elementů v ploše
- Elementy na stránce lze zmenšovat, zvětšovat, otáčet apod.
- Formát zápisů **transform**: **transformaciFunkce**;
- **rotate()** – rotace elementu

```
h1 {  
    transform: rotate(180deg);  
}
```

2D transformace (II)

- **translate()** – posun z výchozí pozice

```
h2 {  
    transform: translate(50px, 100px); /* osa x, osa y */  
}
```

- **skew()** – zkosení elementu

```
h3 {  
    transform: skew(-30deg,5deg); /* v ose x, v ose y */  
}
```

2D transformace (III)

- **scale()** – umožňuje zmenšovat nebo zvětšovat elementy

```
h2 {  
    transform: scaleY(3); /* definuje násobek zvětšení  
    /zmenšení */  
}
```

2D transformace (IV)

- **matrix()** – komplexní způsob transformace
 - Formát: **transform: matrix(a, b, c, d, X, Y)**
 - a – nahrazuje scaleX()
 - d – nahrazuje scaleY()
 - b – nahrazuje skewX() (v radiánech)
 - c – nahrazuje skewY() (v radiánech)
 - X – nahrazuje translateX()
 - Y – nahrazuje translateY(),

```
h4 {  
    transform: matrix(1, 0.577350269, 0, 1, 100, 500);  
}
```

2D transformace (V)

- **transform-origin** – definuje bod, od kterého se odvíjí umístění transformovaného elementu

```
h2 {  
    transform-origin: left top;  
}
```

- **Mnohonásobná transformace**

```
div {  
    transform: rotate(45deg) scale(2) translate(50px, 50px);  
}
```

3D transformace

- Umožňuje trasformaci elementu v prostoru
- **translate3d(x, y, z)**
- **scale3d(x, y, z)**
- **rotate3d(x, y, z, angle)**
- **perspective(n)**
- **matrix3d(. . .)**

Přechody

- Umožňují animovat přechod elementů z jednoho stavu do druhého
- Přechody se například používá při najetí myši nad prvek apod.
- Jednotlivé části **transition**:
 - **transition-property** – definice vlastností, které se mají animovat
 - **transition-duration** – čas přechodu
 - **transition-timing-function** – časový průběh (konstantní, proměnlivý, atd.)
 - **transition-delay** – zpoždění než začne přechod
- Definice jedním příkazem:

```
transition: property duration timing-function delay;
```

Přechody (II)

- Příklad:

```
div {  
    color: black; background: white;  
    transition: all 2s linear 1s;  
}  
  
div:hover {  
    color: green; background: blue;  
}
```

- Řešení: se zpožděním 1s dojde ve 2s (s lineárním průběhem) ke změně všech vlastností.

Přechody (III)

- Příklad:

```
div {  
    transform-origin: center;  
    transform: none;  
    transition: transform 2s ease;  
}  
div:hover {  
    transform: rotate(-30deg);  
}
```

- Viz specifikace [CSS3 Transitions](#).

Animace

- Umožňují vytvářet skutečné animace
- Definice jednotlivých fází
- Mnohé vlastnosti jsou obdobné jako v modulu přechodů (doba trvání, průběh animace, atd.)
- Jednotlivé snímky se definují pomocí **@keyframes**

```
@keyframes nazev_animace {  
    from { background: black; }  
    ...  
    50% { background: blue; }  
    ...  
    to { background: white; }  
}
```

Vlastnosti animací

- **animation-name** – jméno předdefinované animace
- **animation-duration** – délka průběhu animace
- **animation-timing-function** – definice průběhu času
- **animation-delay** – zpoždění začátku animace
- **animation-iteration-count** – počet opakování animace
- **animation-direction** – definice způsobu přehrávání
- Definice jedním příkazem:

```
animation: animation-name animation-duration animation-
timing-function
          animation-delay animation-iteration-
count animation-direction;
```

Vlastnosti animací (II)

- **animation-play-state** – možnost přerušení animace
- Příklad:

```
a:hover {  
    animation-play-state: paused;  
}
```

Animace – příklad

```
@keyframes animace {  
    from { background: green; }  
    50% { background: yellow; }  
    to { background: red; }  
}  
  
a {  
    animation: animace 10s linear 2s 5 alternate;  
}  
  
a:hover {  
    animation-play-state: paused;  
}
```

Animace – příklad (II)

- Mnohonásobná animace

```
@keyframes anim { ... }  
@keyframes anim2 { ... }  
  
a {  
    animation: anim 6s linear 2s 5, anim2 3s 0.5s 1;  
}
```

- Chování: animace **anim**, bude trvat 6s (lineární průběh). Začne po dvou sekundách a bude 5 krát přehrána. Zároveň bude spuštěna animace **anim2**, která bude trvat 3s. Začne s 0.5s zpožděním a bude přehrána jen jednou.

Další vlastnosti CSS3

- Spousta vlastností není v prohlížečích ještě implementována
- **Layoutovací moduly**
 - Flexible Box Layout (IE, FF, Chrome)
 - Template layout
 - Grid layout (IE)
 - Region (Chrome)
 - Grid position (IE)
- **Speech** – definice chování čtečky textu (Opera)
- atd.

respond.js

- Knihovna na podporu min/max-width CSS3 Media Queries pro IE8 a starší prohlížeče
- respond.js se musí vyskytovat jako první JS v dokumentu
- respond.js musí následovat po CSS souborech
- [Oficiální stránky](#)

Modernizr

- JavaScript knihovna detekující HTML5 a CSS3 vlastnosti v uživatelském prohlížeči
- Automatická inicializace objektu knihovny
- Snadné použití

```
<script src="modernizr.min.js"></script>
<script>
if (Modernizr.canvas) {
    // canvas je podporován
} else {
    // canvas není podporován
}
</script>
```

Odkazy

- CSS3 na webu W3C
<http://www.w3.org/Style/CSS/>
- CSS3 in Style
<http://www.w3.org/Talks/2012/0416-CSS-WWW2012/>
- Vzhůru do CSS3
<http://www.vzhurudolu.cz/css3/>
- CSS3 Please!
<http://css3please.com/>
- CSS3 Test
<http://css3test.com/>
- Can I Use?
<http://caniuse.com/>

9. JavaScript a jQuery

Dynamické HTML

JavaScript

- Jazyk pro zápis programů připojených k webovým stránkám
- Zapsaný program se provádí **v prohlížeči**
- Vlastnosti jazyka
 - Vychází z jazyků C a Java
 - Interpretovaný
 - Objektově orientovaný
 - Case sensitive (citlivý na velká a malá písmena)
- JavaScript nemá nic společného s Javou (i když je syntakticky podobný)

Použití JavaScriptu

- Dynamický obsah
 - Generování obsahu dokumentu podle různých okolností
 - Např. vložení aktuálního času, reakce na verzi prohlížeče, ...
- Interaktivní práce s dokumentem
 - Reakce na události (pohyb myši, kliknutí, změna hodnoty, ...)
 - Změna vzhledu prvků dokumentu
- Ovládání prohlížeče
 - Pohyb v historii
 - Otevírání oken, stavový řádek, ...

Vložení JavaScriptu

- Dvě varianty - vložený a externí skript

```
<script type="text/javascript">  
... kód skriptu ...  
</script>
```

nebo

```
<script type="text/javascript" src="program.js">  
</script>
```

Vykonání skriptu

- Skript může být vložen v hlavičce i těle dokumentu
- Vykoná se v okamžiku, kdy na něj prohlížeč narazí

```
<html>
<head><title>Titulek</title></head>
<body>
Teď se spustí skript<br>
<script type="text/javascript">
    document.write("Už se <em>spustil</em><br>");
</script>
Zpracuje se zbytek dokumentu.
</body>
</html>
```

Příklad

Ted' se spustí skript
Už se ***spustil***
Zpracuje se zbytek dokumentu.

Generování výstupu

- Nejčastěji generujeme HTML kód

```
var a = 0;
while (a < 10) {
    document.write("Řádek " + a + "<br>");
    a++;
}
```

Příklad

Řádek 0
Řádek 1
Řádek 2
Řádek 3
Řádek 4
Řádek 5
Řádek 6
Řádek 7
Řádek 8
Řádek 9

Cykly – generování tabulky

```
var radek, sloupec;
for (radek = 1; radek <= 10; radek++)
{
    document.write("<tr>");
    for (sloupec = 1; sloupec <= 10; sloupec++)
    {
        document.write("<td>" + (radek*sloupec) + "</td>");
    }
    document.write("</tr>");
}
```

Příklad

1	2	3	4	5	6	7	8	9	10
2	4	6	8	10	12	14	16	18	20
3	6	9	12	15	18	21	24	27	30
4	8	12	16	20	24	28	32	36	40
5	10	15	20	25	30	35	40	45	50
6	12	18	24	30	36	42	48	54	60
7	14	21	28	35	42	49	56	63	70
8	16	24	32	40	48	56	64	72	80
9	18	27	36	45	54	63	72	81	90
10	20	30	40	50	60	70	80	90	100

Aplikační rozhraní v prohlížeči

- Objekty reprezentující prohlížeč
 - **window** - okno prohlížeče
 - další vnořené objekty
- Model dokumentu – DOM
 - **document** - dokument jako celek
 - **element** - element HTML kódu

Objekt window

- Vnořené objekty (jako vlastnosti)
 - **window.location** - adresa zobrazeného dokumentu
 - **window.history** - práce s historií
 - **window.screen** - obrazovka počítače
 - **window.document** - aktuálně zobrazený dokument
- Ostatní vlastnosti
 - **window.defaultStatus** - stavový řádek
 - **window.defaultStatus = "Nějaký text";**
 - **window.status** - stavový řádek (dočasný)
 - **window.status = "Nějaký text";**

Standardní dialogy

- Okno se zprávou: `window.alert()`

```
window.alert("Nějaká zpráva");
```

Test

- Vstupní okno: `window.prompt()`

```
var x;  
x = window.prompt("Zadej jméno:", "nikdo");  
window.alert("Zadal jsi: "+x);
```

Test

Objekt `window.location`

- Adresa aktuálně zobrazené stránky
- Zobrazení jiného dokumentu:
 - `window.location.href = "http://www.google.com";`
- Nové natažení dokumentu (`reload`):
 - `window.location.reload(true);` – vždy
 - `window.location.reload(false);` – pouze při změně
- Zjištění aktuální adresy
 - `adresa = window.location.href;`
 - `domena = window.location.hostname;`

Funkce

- Definice funkce:

```
function secti(a, b)
{
    return a + b;
}
```

- Volání funkce:

```
<script type="text/javascript">
    document.write("Výsledek je: " + secti(12, 6));
</script>
```

Události

- Událost je vnější okolnost, která nastává
 - Činností prohlížeče
 - Dokument byl načten, ...
 - Činností uživatele
 - Událost myši (pohyb, stisk tlačítka, ...)
 - Událost klávesnice (stisk nebo puštění tlačítka, ...)
 - Manipulace s prvky formuláře (změna hodnoty, stisk tlačítka, ...)
 - Každé události může být přiřazen JavaScriptový kód, který se provede
 - Pro každý element jiný

Definice událostí

- Každá událost má jméno
 - Např. kliknutí myši: **onclick**
- Každý element může mít atribut se stejným jménem, který obsahuje kód v JavaScriptu

```
<span onclick="window.alert('kliknuto');">  
nějaký text  
</span>
```

- Ne všechny elementy mohou reagovat na všechny události
 - Záleží na prohlížeči

Události dokumentu

- **onLoad**

- Při načtení stránky nebo obrázku
- Pouze <body> nebo

- **onUnload**

- Při opuštění stránky
- Pouze <body>

- **onResize**

- Změna velikosti okna
- Pouze <body>

- **onScroll**

- Rolování okna
- Cokoliv co scrolluje

Události myši

- Aplikovatelné na všechny elementy
- **onClick** - kliknutí na prvek
- **onDbClick** - dvojklik
- **onMouseOver** - najetí kurzorem
- **onMouseOut** - opuštění kurzorem
- **onMouseMove** - pohyb myši nad prvkem
- **onMouseDown** - stisk tlačítka
- **onMouseUp** - puštění tlačítka

Události klávesnice

- Pro všechny elementy
- Aplikuje se na **zaměřený** element
- **onKeyPress** - stisknutí klávesy (logické)
- **onKeyDown** - stlačení klávesy
- **onKeyUp** - uvolnění klávesy

Propagace událostí

```
<a href="stranka.html"
    onclick="alert('Kliknuto!'); return true;">Odkaz 1</a>
<br>
<a href="stranka.html"
    onclick="alert('Kliknuto!'); return false;">Odkaz 2</a>
```

Odkaz 1

Odkaz 2

Zpracování pomocí funkcí

- Složitější zpracování události je vhodné implementovat pomocí funkce
 - Přehlednější
 - Znovupoužitelné

funkce.js:

```
function prejdi()
{
    var adresa = window.prompt("Cílová adresa");
    if (adresa != null)
        location.href = adresa;
}
```

Test

Použití v dokumentu

```
<html>
<head>
    ...
    <script type="text/javascript" src="funkce.js">
    </script>
    ...
</head>
<body>
    ...
    <a href="..." onclick="prejdi(); return false;">
        Přejít na stránku</a>
    ...
</body>
</html>
```

Zpracování pomocí funkcí (II)

- Alternativní zápis přímo v JavaScriptu
 - Definujeme tzv. **event handler**

```
<body>
    ...
    <a href="..." id="odkaz">Přejít na stránku</a>
    <script type="text/javascript">
        document.getElementById('odkaz').onclick = prejdi;
    </script>
    ...
</body>
```

- Existují další pokročilé způsoby zpracování událostí

Přístup k prvkům dokumentu

- Staré (nestandardní) rozhraní
 - Obrázky – pole **document.images[]**
 - Formuláře – pole **document.forms[]**
 - Odkazy – pole **document.links[]**
- Standardní rozhraní představuje **DOM model**

Změna obrázku

```
<script type="text/javascript">
function usmevOn()
{
    document.images['face'].src="face1.png";
}
function usmevOff()
{
    document.images['face'].src="face2.png";
}
</script>

```



Použití DOM modelu

- Základní DOM rozhraní je **Element**
- DOM HTML rozhraní **HTMLElement**
 - Popisuje obecný element v HTML kódu
 - Vlastnosti **`id`, `title`, `className`**, ...
 - Pro mnoho elementů další odvozené třídy
- V browserech má rozhraní **Element** mnoho vlastností, které dosud nejsou standardizovány

Získání elementu z dokumentu

- Podle hodnoty **id** (**Document** i **Element**)
 - ```
var el = document.getElementById('hodnotaid');
var titulek = el.title
```
- Podle jména elementu (**Document** i **Element**)
  - ```
polozky = document.getElementsByTagName("input");  
var sum = 0;  
for (var i = 0; i < polozky.length; i++) {  
    if (polozky[i].type == 'text')  
        sum = sum + polozky[i].value;  
}
```
- Podle hodnoty atributu **name** (jen **Document**)
 - **document.getElementsByName()**

Získání elementu z dokumentu (II)

- Podle jména třídy **class** (zavedeno od HTML5)
 - `var nadpisy= document.getElementsByClassName('nadpis');`
- Podle **syntaxe CSS** (zavedeno od HTML5)
 - `var bunky = document.querySelectorAll("table td");`

Změna obsahu

```
<script type="text/javascript">
function pricti()
{
    var e = document.getElementById('cislo');
    var n = e.innerHTML;
    e.innerHTML = Number(n) + 1;
}
</script>

Cislo:
<span id="cislo" onmouseover="pricti();">0</span>
```

Cislo: 0

Změna obrázku

```
<script type="text/javascript">
function usmevOn()
{
    document.getElementById('DOMface').src = "data/face1.png";
}
function usmevOff()
{
    document.getElementById('DOMface').src = "data/face2.png";
}
</script>

```



Příslušná část DOM specifikace

Přístup ke stylům

```
<script type="text/javascript">
function zmen()
{
    var e = document.getElementById('elem');
    e.style.marginLeft="5em";
    e.style.border="1px solid blue";
}
</script>
```

Nějaký element

Nějaký element

Aplikace JavaScriptu

Generování stylů

- Pomocí volání **document.write()** lze generovat libovolnou část dokumentu
- Lze např. generovat styly:

```
<html><head>
  <title>Cookie</title>
  <script type="text/javascript" src="style.js">
    </script>
</head>
<body>
  ...
</body>
```

Generování stylů

- Chceme skript, který vygeneruje následující:

```
<style type="text/css">  
... nějaká CSS pravidla ...  
</style>
```

Příklad

Soubor style.js

```
document.write('<style type="text/css">');

date = new Date();

var pozadi;
if (date.getHours() >= 12)
    pozadi = "blue";
else
    pozadi = "green";

document.write('body { background-color: '+pozadi+'; }');
document.write('</style>');
```

Cookies

- Cookie - malý objem dat, která si stránka může uložit do prohlížeče
- Každý cookie má svoje ***jméno*** a ***hodnotu***
- Každá stránka se dostane jen ke svým hodnotám
- Max. 20 cookies z jedné domény, max. 4kB každý
- Použití: různá uživatelská nastavení
- Uživatel může zakázat ukládání cookies

Cookies v JavaScriptu

- Vlastnost **document.cookie**
- Zápis:
 - **document.cookie = "JMEN0=hodnota";**
 - Hodnotu je vhodné zpracovat **escape()**
 - Např. **document.cookie = "JAZYK="+escape("čeština");**
- Čtení:
 - Vlastnost **document.cookie** obsahuje seznam cookies oddělených středníky
 - Např. **JAZYK=cesky; JMEN0=Jan; PRIJMENI=Novak**
 - Správná hodnota se musí najít pomocí **indexOf()** a **substring()**

Trvanlivost cookies

- Standardně se cookie maže se zavřením prohlížeče
- Lze zadat trvanlivost, dokdy má platit:
 - **document.cookie = "JAZYK=hodnota; EXPIRES="+datumGMT;**

```
cas = new Date();
cas.setTime(cas.getTime() + 1000 * 120);
document.cookie = "MOJE=hodnota; EXPIRES="+cas.toGMTString();
```

Příklad

Knihovna jQuery

jQuery

- JavaScript framework orientovaný na práci s HTML dokumentem
 - Snazší přístup k elementům a manipulace
 - Řeší problémy s kompatibilitou prohlížečů
- Integrace jQuery

```
<script type="application/javascript"
       src="http://ajax.googleapis.com/ajax/libs/
             jquery/1.11.2/jquery.min.js">
</script>
```

- Alternativně např. jQuery CDN

Použití jQuery

- Funkce **jQuery(selektor)**
 - Najde v HTML kódu všechny elementy, které odpovídají selektoru
 - Vytvoří objekt, který reprezentuje nalezené elementy
 - Možno zapsat i pomocí **\$(selektor)**

```
var e = $('div.article');
var pocet = e.length;
```

- Vrácený objekt umožňuje manipulaci s příslušnými HTML elementy

Událost Document ready

- Je vyvolána v okamžiku načtení dokumentu
- V těle se obvykle inicializují události elementů
- Příklad ekvivalentního kódu:

```
$(document).ready(function() {
    // JavaScript kód ...
});
```

```
$(function() {
    // JavaScript kód ...
});
```

Styl elementů

- Přidání (odejmutí) třídy elementu

```
$('p').addClass('odstavec');  
$('p strong').removeClass('vyrazne');  
$('#menu').toggleClass('hidden');
```

- Přímá změna stylu

```
if ($('#msg').css('color') == 'black')  
{  
    $('#msg').css('color', '#fa0000');  
}
```

Obsah elementů

- Získání obsahu elementu:
 - **text()** – získání textového obsahu zvoleného elementu
 - **html()** – získání obsahu zvoleného elementu (včetně HTML značek)
 - **val()** – získání hodnoty z formulářového elementu
- Nastavení obsahu se děje stejnými metodami jako jejich získání (pouze se přidá parametr s hodnotou)

```
$( 'p' ).text( 'Nastavený obsah' );
```

Hodnoty atributů

- Získání hodnoty atributu –
`$(selektor).attr('nazev_atributu')`
- Nastavení hodnoty atributu –
`$(selektor).attr('nazev_atributu', 'hodnota')`
- Příklad:

```
$("button").click(function() {  
    $('a#fit').attr('href', 'http://www.fit.vutbr.cz');  
});
```

Vložení nového HTML obsahu

- **append()** – vložení na konec vybraného elementu
- **prepend()** – vložení na začátek vybraného elementu
- **after()** – vložení za vybraný element
- **before()** – vložení před vybraný element
- Příklad:

```
$("button").click(function() {  
    var obsah = '<span>Nový obsah <strong>aktualizace</strong>  
<span>';  
    $("p#newObsah").append(obsah);  
});
```

Smazání obsahu

- **remove()** – smazání vybraného elementu (včetně jeho potomků)

```
$("p#obsah").remove();
```

- **empty()** – smazání potomků vybraného elementu

```
$("p#obsah").empty();
```

Efekty

```
$('#test').hide();
$('#test').show();
$('#test').toggle();
```

Skrýt Zobrazit Přepnout

Testovací
element

Další obsah

Efekty

```
$('#test').hide('slow');  
$('#test').show('slow');  
$('#test').toggle('slow');
```

Skrýt Zobrazit Přepnout

Testovací
element

Další obsah

Efekty

```
$('#test').fadeIn('slow');  
$('#test').fadeOut('slow');  
$('#test').fadeToggle('slow');
```

Skrýt Zobrazit Přepnout

Testovací
element

Další obsah

Efekty

```
$('#test').slideUp('slow');  
$('#test').slideDown('slow');  
$('#test').slideToggle('slow');
```

Skrýt Zobrazit Přepnout

Testovací
element

Další obsah

Obsluha událostí

```
$('.link1').click(function() {  
    $('#test').slideUp('slow');  
});  
...
```

[Skrýt](#) [Zobrazit](#) [Přepnout](#)

Testovací
element

Další obsah

Obsluha událostí (II)

```
$('.link1').mouseout(function() {  
    $('#test').slideUp('fast');  
});  
$('.link1').mouseover(function() {  
    $('#test').slideDown('fast');  
});
```

Přepnout

Testovací
element

Obsluha událostí (III)

- Další události:
 - **keypress**, **keydown**, **keyup**
 - **submit**, **change**, **focus**, **blur**
 - **load**, **resize**, **scroll**, **unload**
 - atd.

Callback funkce

- Funkce je vyvolána v okamžiku dokončení požadovaného efektu
- Syntaxe: **`$(selektor).hide(rychlosť,callback);`**
- Příklad:

```
$("a").click(function() {
    $("p").hide("slow",function() {
        alert("Odstavec je již skryt");
    });
});
```

Řetězení metod

- Příklad:

```
$("button").click(function() {  
    $("#p1").css("color","blue").slideUp(2000).slideDown(2000)  
});
```

Animovaný text!!

Click me

Spousta dalších užitečných vlastností

- Podpora asynchronního zpracování dokumentu – AJAX
- Obrovské množství knihoven – jQuery UI
 - Dialogová okna
 - Kalendáře
 - Záložkování
 - Progress bary
 - atd.
- <http://jqueryui.com>

Odkazy

- JavaScript na quirksmode.org quirksmode.org
- Gecko DOM reference
- W3C DOM
 - W3C DOM Level 2 HTML
- jQuery
 - Aplikační rozhraní
 - jQuery UI

10. Informační architektura webu, použitelnost, přístupnost

Informační architektura webu

Informační architektura

- **Organizace informací**
 - Jak jsou data logicky organizována
- Označování (reprezentace) informací
 - Dodávají kontext prezentovaným informacím
- Navigační systémy
 - Jak se bude uživatel na webu orientovat
- Vyhledávací systémy
 - Způsoby vyhledávání obsahu

Organizační systém

- Organizační schéma
 - Podle čeho informace kategorizujeme
- Organizační struktura
 - Jakým způsobem jednotlivé kategorie organizujeme

Organizační schéma

- **Exaktní schémata** - musíme přesně vědět, co hledáme
 - Abecední
 - Chronologické
 - Geografické
- **Nejednoznačná schémata**
 - Tématické - kategorie článků, druhy zboží, ...
 - Podle činností - čeho chce uživatel dosáhnout
 - Podle rolí návštěvníků - např. web obce: obyvatel, turista, novinář, ...
 - Metaforické - např. organizujeme web jako obchodní dům (vchod, informace, oddělení, ...)
 - ... nebo kombinace předchozích

Organizační struktury

- **Hierarchické**
 - Snadno pochopitelné a zapamatovatelné
 - Snadná navigace
- **Databázové**
 - Rovnocenné dokumenty zařazené podle metadat
 - Např. osobní stránky uživatelů
- **Obecný graf**
 - Hypertextové odkazy mezi daty navzájem
 - Obtížné na orientaci

Informační architektura

- Organizace informací
 - Jak jsou data logicky organizována
- **Označování (reprezentace) informací**
 - Dodávají kontext prezentovaným informacím
- Navigační systémy
 - Jak se bude uživatel na webu orientovat
- Vyhledávací systémy
 - Způsoby vyhledávání obsahu

Způsoby označení informací na webu

- Nadpisy (popisky)
 - Mohou tvořit hierarchii
- Kontextové odkazy
 - Volba textu odkazu
- Navigační systém
 - Označení položek - hlavní stránka, vyhledávání, ...
- Ikony
 - Vylepší vzhled stránky
 - Pouze omezený počet rozlišitelných ikon

Důležité vlastnosti

- Specifičnost
 - Vyhnut se příliš obecným označením
- Konzistence
 - **Stylu** - interpunkce, malá a velká písmena
 - **Prezentace** - fonty, barvy
 - **Syntaxe** - nazývání kapitol podstatnými jmény, otázkami
 - **Granularity** - označování na stejném úrovni abstrakce
 - **Komplexnosti** - logicky související informace na stejném místě
 - **Terminologie** - odborné vs. obecné výrazy

Informační architektura

- Organizace informací
 - Jak jsou data logicky organizována
- Označování (reprezentace) informací
 - Dodávají kontext prezentovaným informacím
- **Navigační systémy**
 - Jak se bude uživatel na webu orientovat
- Vyhledávací systémy
 - Způsoby vyhledávání obsahu

Druhy navigace

- Hlavní typy navigace
 - **Globální** - v které části webu se nacházíme
 - Viditelná stále
 - **Lokální** - co je v této části (např. kapitole)
 - Liší se v jednotlivých sekcích, konzistence
 - **Odkazy v textu**
 - Hypertextové odkazy na další související informace
- Pomůcky
 - **Mapa stránek**
 - **Index**
 - **Návod** - lineární, krok za krokem

Tvorba kontextu

- Uživatel by měl vždy vědět
 - Na jakých stránkách se nachází
 - V jaké části se nachází
- Vyhledávač může uživatele poslat „doprostřed“ webu
- Měla by vždy být jasná cesta
 - Na hlavní stránku
 - Na začátek sekce
 - O úroveň výše
 - Z kořenové stránky na tuto stránku

Informační architektura

- Organizace informací
 - Jak jsou data logicky organizována
- Označování (reprezentace) informací
 - Dodávají kontext prezentovaným informacím
- Navigační systémy
 - Jak se bude uživatel na webu orientovat
- **Vyhledávací systémy**
 - Způsoby vyhledávání obsahu

Vyhledávání

- Není nutné vždy
- Implementačně i provozně náročné
- Hodí se pro weby:
 - S velkým množstvím obsahu
 - S rychle se měnícím obsahem
 - Které mají nepřehlednou standardní navigaci
 - U kterých to uživatelé očekávají
- Hledat lze
 - V obsahu dokumentů
 - V metadatech (náročnější, přesnější)

Vlastnosti vyhledávacích systémů

- Rozhraní
 - Vyhledávací box
 - Pokročilé vyhledávání
- Specifikace vyhledávání
 - Booleovské operátory
 - Specifikace kritérii (omezení prohledávaných dokumentů)
 - Volba prezentace výsledků (třídění, podrobnost)

Použitelnost webu

Použitelnost webu

- Jedno z měřítek kvality zpracování webu
- Udává míru „uživatelského komfortu“ při používání stránek
- Dobře použitelná stránka
 - Návštěvník se v ní lehce orientuje
 - Rozumí jí
 - Nenutí uživatele přemýšlet, je ***intuitivní***

Zhruba **40% návštěvníků** se již na stránky nevrátí, pokud jejich prvotní zkušenosti byla negativní. Přibližně **50% obchodů je ztraceno**, pokud návštěvníci dostatečně rychle nenaleznou požadované informace.

- Jakob Nielsen a Forrester Research, (*Failure of Corporate Websites, useit.com, 18.10.1998*)

Význam použitelnosti

- U komerčních webů znamená použitelnost **zisk**
 - Špatná použitelnost může zákazníka odradit od objednávky
- Správně provedené **testy použitelnosti** mohou odhalit překážky při hledání konkrétních informací na stránkách

Studie o chování uživatelů na webu zjistily nízkou toleranci složitých designů a pomalých webů. Lidé **nechtějí čekat a nechtějí se učit**, jak používat webovou stránku.

- Jakob Nielsen

Faktory použitelnosti

- Rychlosť učení
 - Jak rýchle nový uživatel zvládne základní úlohy?
- Efektivita používání
 - Jak efektivně zkušený uživatel dokáže plnit zadané úkoly?
- Zapamatovatelnost
 - Je snadné si zapamatovať řešení konkrétního úkolu?
- Odolnost proti chybám
 - Jak snadné je udělat chybu a jak snadno lze chybu napravit?
- Subjektivní uspokojení
 - Používá uživatel váš web ***rád***?

Použitelnost vs. přístupnost

- Často zaměňované oblasti
- **Přístupnost** - je pro uživatele obsah stránek **technicky dostupný?**
- **Použitelnost** - je uživatel schopen získaný obsah k něčemu konkrétnímu využít?

Přístupnost je nutnou avšak nikoliv postačující podmínkou použitelnosti webu

Návrh použitelného webu

1. Plánování
 - Čeho chceme dosáhnout?
2. Sběr dat od uživatelů
 - Jací uživatelé budou web využívat
 - Co od něj budou očekávat
3. Tvorba prototypu
 - Předběžná verze webu určená pro testování
4. Sběr, psaní a revize obsahu
 - Selekce, organizace a forma prezentace obsahu
5. Testování použitelnosti
 - Nejlépe na skutečných uživatelích
6. Propagace webu a údržba

Fáze 1 - plánování

- Plánování pomáhá přesně vymezit cíle projektu
- Nejdříve je třeba si ujasnit
 1. Proč vytváříme web
 2. Kdo by měl web využívat
 3. Kdy a proč by jej měl využívat

Proč vytváříme web?

- Návrh měřitelných a kontrolovatelných cílů
- Je dobré zodpovědět následující otázky
 - Jak poznáme (kvantitativně), že je web úspěšný?
 - Jaké budou důsledky nebude-li web úspěšný?
- Vyhnut se příliš obecným cílům
 - Např. „informovat o našich produktech“
- Lépe je spojit cíle s obchodními potřebami
 - Např. „zvýšení prodeje horských kol o 30%“
 - „Snížení zátěže telefonické uživatelské podpory o 50%“

Kdo by měl web využívat?

- Seznam cílových skupin
- Např. základní rozdělení pro elektoronický obchod:
 - Uživatelé zjišťující informace o zboží (parametry, cenu, ...) „browsers”
 - Uživatelé mající v úmyslu nakoupit konkrétní produkt „buyers”
- Závisí na zaměření webu, např. web státní instituce:
 - Občan, podnikatel, novinář, ...
- Vypracování charakteristik každé skupiny
 - Např. zaměstnaný, hledá detailly, znalý oboru, malé zkušenosti s WWW
 - Jedná se o odhad, ověří se ve fázi testování
- Tyto charakteristiky musí být respektovány při návrhu webu.

Kdy a proč uživatelé přicházejí

- Uživatel přichází s nějakým vlastním cílem
- Tvorba typických scénářů:
 - Např. Karlovi dosloužil televizor a potřebuje co nejrychleji koupit nový obdobných parametrů, jako jeho starý přístroj.
 - Josef vyhrál ve Sportce a chce si pořídit televizor špičkových parametrů, neví ale přesně, jaké parametry to jsou.
- Další scénáře lze získat přímo od uživatelů v rámci dalších kroků

Fáze 2: sběr dat od uživatelů

- Nejlepší cestou získání informací o uživatelích je
 - Zeptat se jich
 - Pracovat s nimi
- Chceme získat informace o tom
 - Jaké informace potřebují
 - Jakým způsobem o informacích přemýšlí, a jak je organizují
 - Co od webu očekávají
 - Jaká je úroveň jejich znalostí o předmětu
 - Jak zkušení jsou v užívání webu
- Můžeme získat mnoho reálných scénářů užití webu

Techniky získávání dat od uživatelů

- **Předběžný test použitelnosti**

- Test na stávajícím (starém webu), pokud je k dispozici, nebo na konkurenčním webu
- Malá skupina „pokusných“ uživatelů (5 - 12)
- Sledujeme chování uživatelů při plnění úkolů
- Možno i vzdáleně (analýza logů)

- **Kontextuální interview**

- Sledujeme uživatele v prostředí, kde obvykle pracuje
- Obvykle nezadáváme zvláštní úlohy
- Charakteristiky prostředí
 - Technické podmínky
 - Atmosféra pracoviště
 - Má uživateli kdo poradit?

Techniky získávání dat od uživatelů (II)

- **On-line průzkumy**

- Výzkum pomocí webových dotazníků
- Uživatelé vyplňují údaje sami o sobě
- Důležité parametry:
 - Jaké informace chceme získat
 - Kde se ženeme respondenty
 - Dotazník by měl být krátký (do 10 otázek)
 - Časově nenáročný na vyplnění
 - Kombinace otázek na uživatele (demografické údaje) a na jeho preference

Techniky získávání dat od uživatelů (III)

- **Individuální interview**

- Menší počet respondentů (kolem 15)
- Nesledujeme chování, pouze zjišťujeme očekávání, a preference
- Obvykle rozhovor na základě předem dané osnovy
- Rozhovor by měl vést zkušený odborník

- **Skupinová diskuse**

- Skupina 8 - 12 respondentů
- Zjišťujeme
 - Postoje a požadavky uživatelů
 - Reakce a nápady na prototypy
- Nejlepší je opět zkušený moderátor

Techniky získávání dat od uživatelů (IV)

- **Card sorting** (třídění kartiček)
 - Uživatel vstupuje do procesu organizace dat na webu
 - Uživatel dostane sadu karet s tématy obsahu webu
 - První kartu položí na stůl
 - U každé další karty se rozhodne, zda náleží do stejné kategorie, nebo si zaslouží vytvoření nové kategorie
 - Může přeskupovat karty
 - Uživatel by měl „přemýšlet nahlas“

Fáze 3: tvorba prototypu

- Prototyp je „atrapa“ budoucího webu
- Obvykle neobsahuje veškerý obsah
- Obsahuje ukázku domácí stránky a několika stránek různých typů
- Cílem je mít ukázku informační architektury, navigace a designu pro testování na uživatelích
- Umožňuje ověřit návrh webu před investicí času a prostředků do tvorby celého webu
- Měl by být snadno adaptovatelný podle zjištěných skutečností

Fáze 4: tvorba obsahu

- Selekce obsahu
 - Je každá informace relevantní k obsahu webu?
 - Potřebuje uživatel opravdu danou informaci?
- Organizace obsahu
 - Obsah musí být „snadno použitelný“
 - Podán srozumitelným jazykem
 - Přehlednou formou
 - Logicky organizovaný

Logická organizace obsahu

- Členění obsahu na snadno zvládnutelné části
 - Obsah nesmí čtenáře odradit délkou a složitostí
 - Krátké odstavce a věty
 - Seznamy, tabulky, obrázky
 - Příklady
- Časté použití nadpisů
 - Zjednodušují orientaci na stránce
 - Stránka vypadá zajímavěji
 - Každá krátká část, tabulka má mít nadpis
- Odkazy na důležité nadpisy na začátku (záložky)
 - V případě delších dokumentů
 - Pomáhají v orientaci

Logická organizace obsahu (II)

- Sekvenční informace je třeba prezentovat ve správném pořadí
 - Např. k pochopení jedné části je třeba znát jinou část webu
- Pokud pořadí nehraje roli, upřednostníme informace, které jsou pro uživatek důležitější

Prezentace obsahu

- Práce s volným místem na stránce
 - Informace by se měly přirozeně shlukovat
 - Příliš volného místa na stránce působí prázdně
 - Málo volných míst - informace splývají a netvoří „shluky“
- Efektivní vyjadřování
 - Každá věta musí mít význam, žádná „prázdná slova“
 - Opakované revize s časovým odstupem

Prezentace obsahu (II)

- Používaný jazyk
 - Web by měl mluvit jazykem uživatele
 - Vyvarovat se odborného „žargonu“
- Seznamy zpřehledňují obsah
 - Odrážky jsou pochopitelnější než dlouhé věty
 - Pro jednotlivé kroky postupu je vhodný číslovaný seznam
- Tabulky
 - Přehlednější než slovní popis
 - Např. cena pro jednotlivce, cena pro firmy, ...

Prezentace obsahu (III)

- Příklady
 - Uživatel často upřednostní příklad před dlouhým čtením teorie
- Vžité formy prezentace
 - Mnoho údajů je zvykem prezentovat určitým způsobem
 - Např. adresu, cenu, ...
- Ikony
 - Oživují stránku
 - Zdůrazňují význam slov (např. telefon, ...)
- Časté využití obrázků (kde je to vhodné)

Uživatelé na webu

- Typický uživatel spěchá
 - Hledá konkrétní informaci
 - Má spoustu další práce
 - Stránek k prohlédnutí je mnoho
- Uživatelé na webu ***nečtou***
 - Pouze „skenují“ obsah dokumentu
 - Když dokument nevypadá relevantně, hledají jiný
- Čtení na monitoru může být těžší
 - Lépe prezentovat podstatné informace s odkazem na celý dokument
 - Tiskové verze důležitých dokumentů

Fáze 5: testování použitelnosti

- Cíle testování
 - Zvládnou uživatelé danou úlohu?
 - Jak rychle?
 - Jsou s touto rychlostí spokojeni?
 - Jakými cestami dosahují výsledku?
 - Připadají jim tyto cesty efektivní
 - Jaká jsou nejčastější problematická místa?
 - Zkouší postupy, které web neumožňuje?

Fáze 6: propagace a údržba

- Web musí být k nalezení
 - Umístění odkazů na jiné weby
 - Atraktivita obsahu zaručí častější odkazování
 - Optimalizace pro vyhledávače
 - Tiskové zprávy
 - Časté aktualizace = návštěvníci se častěji vrací
- Údržba webu
 - Aktualizace obsahu
 - Kontrola odkazů
- Výhledově další redesign
 - Mohou se změnit požadavky, technologie nebo uživatelé

Přístupnost webu

Přístupný web

- K přístupu k obsahu není zapotřebí konkrétní technické nebo programové vybavení, znalosti či schopnosti
- Až 30% uživatelů webu je nějakým způsobem omezeno (nemají optimální technické vybavení, zobrazovací možnosti, zdravotní stav či správné zkušenosti)

Minoritní skupiny uživatelů

- Uživatelé minoritních operačních systémů a prohlížečů
 - Ne každý má k dispozici Internet Explorer
 - Už vůbec ne Word nebo Excel!
- Uživatelé jiných zobrazovacích zařízení
 - PDA, Smartphones
- Majitelé zastaralých počítačů
 - Starší verze software
 - Malé rozlišení
 - Špatné nebo žádné barvy

Minoritní skupiny uživatelů

- Dyslekci
 - Nečtou špatně strukturované a dlouhé texty
- Barvoslepí (i částečně)
 - Špatně vnímají malé kontrasty
 - Např. odkazy odlišené barevným odstínem
- Slabozrací
 - Vyžadují možnost většího písma
- Tělesně postižení
 - Např. nemohou používat myš

Pravidla pro tvorbu přístupného webu

- Různé organizace vydávají vlastní doporučení
- Mezinárodní
 - W3C - Web Content Accessibility Guidelines 2.0 (WCAG)
- Americké
 - Section 508 - Pravidla pro americkou administrativu
- České
 - SONS - Blind friendly web
 - Zákon č. 365/2000 Sb. o informačních systémech veřejné správy
- <http://www.pravidla-pristupnosti.cz/>

Pravidla pro státní správu

- Od 1.1.2008 závazné pro instituce veřejné správy
 - Vyplývá z novely zákona č. 365/2000 Sb., o informačních systémech veřejné správy
- Většina pravidel je dobře použitelná i pro ostatní weby
- Výrazně zvýší kvalitu prezentace

Vlastnosti přístupného webu

- A. **Obsah stránek je dostupný a čitelný**
- B. Práci s webovou stránkou řídí uživatel
- C. Informace jsou srozumitelné a přehledné
- D. Ovládání webu je jasné a pochopitelné
- E. Kód je technicky způsobilý a strukturovaný
- F. Prohlášení o přístupnosti webových stránek

Obsah stránek je dostupný a čitelný

- Všechny obrázky mají atribut **alt**
- Taktéž **<area>** a **<input type="image">**
- Obrázky bez významu (dekorace) mají prázdný **alt**
- Pro delší popisy je možno použít atribut **longdesc** obsahující URL podrobného popisu

Obsah stránek je dostupný a čitelný

- Textová varianta zvuku případně titulky u videa

Obsah stránek je dostupný a čitelný

- Všechny ovládací prvky funkční i bez doplňkových technologií
- Nejlepší jsou pouze odkazy (fungují vždy)
- CSS a JavaScript použít pouze k jejich vylepšení

Obsah stránek je dostupný a čitelný

- Odkazy typu „červený text vpravo“
- Nejlépe řešit sémantickými značkami
- Možnost využití odkazů v rámci dokumentu

Obsah stránek je dostupný a čitelný

- Barva není jediný navigační prvek
- Lze rozpozнат funkci všech ovládacích prvků
 - **Zejména odkazy**
 - Přidat podtržení, ikonu apod.
 - Nebo funkce vyplývá z kontextu (menu)
- Funkce vysoký kontrast ve Windows

Obsah stránek je dostupný a čitelný

- Kontrast lze měřit, např.
<http://juicystudio.com/services/luminositycontrastratio.php>
- Písmo do 18pt (tučné do 14pt): min. hodnota kontrastu (contrast ratio) **5:1**
- Větší písmo: min. hodnota kontrastu (contrast ratio) **3:1**

Obsah stránek je dostupný a čitelný



- MSIE (<=6) neumí zvětšit písma zadané absolutní velikostí
- Nepoužívat jednotky pt, pc, in, cm, mm, px

Vlastnosti přístupného webu

- A. Obsah stránek je dostupný a čitelný
- B. **Práci s webovou stránkou řídí uživatel**
- C. Informace jsou srozumitelné a přehledné
- D. Ovládání webu je jasné a pochopitelné
- E. Kód je technicky způsobilý a strukturovaný
- F. Prohlášení o přístupnosti webových stránek

Práci s webovou stránkou řídí uživatel

- Ne každý uživatel používá pro výstup standardní obrazovku
 - PDA, hlasové čtečky, řádkový výstup, ...
- Není možno spoléhat na formátování stránky („červený text v pravém sloupci“)
- Nelze předpokládat barvy, možnost tisku, zvukový výstup
- Různá vstupní zařízení
 - Např. chybějící klávesy, jiné přiřazení funkcí klávesám
 - Možná nedostupnost myši

Práci s webovou stránkou řídí uživatel

- Nevynucovat konkrétní prohlížeč či OS
- Vyhnut se formátům vyžadujícím speciální aplikace (DOC, XLS, ...)
- Je-li třeba speciální software, je třeba uvést, jak jej získat (odkaz ke stažení)

Práci s webovou stránkou řídí uživatel

- Nedochází k samovolnému obnovování a přechodu na nový dokument bez akce uživatele, u které se to očekává (kliknutí na odkaz apod.)
- Akce, u kterých se to neočekává (např. pohyb myši) nesmí vést k celkovému přeformátování stránky

Práci s webovou stránkou řídí uživatel

- Aktivace odkazu, odeslání formuláře, ...
- Upozornění např. pomocí atributu **title** nebo textem v odkazu či vedle odkazu

Práci s webovou stránkou řídí uživatel

- Periodické změny tvaru, barvy, ...
- Nejlépe vůbec nepoužívat
- Může mít zdravotní následky
- Nepoužívat hodnotu **blink** v CSS

Práci s webovou stránkou řídí uživatel

- Zvuky obtěžují uživatele
- Vypnutí musí být možné přímo na stránce (ne nastavením OS)

Práci s webovou stránkou řídí uživatel

- Např. přesměrování, potvrzení akce, ...
- Musí být jasné, kolik času zbývá

Vlastnosti přístupného webu

- A. Obsah stránek je dostupný a čitelný
- B. Práci s webovou stránkou řídí uživatel
- C. **Informace jsou srozumitelné a přehledné**
- D. Ovládání webu je jasné a pochopitelné
- E. Kód je technicky způsobilý a strukturovaný
- F. Prohlášení o přístupnosti webových stránek

Informace jsou srozumitelné a přehledné

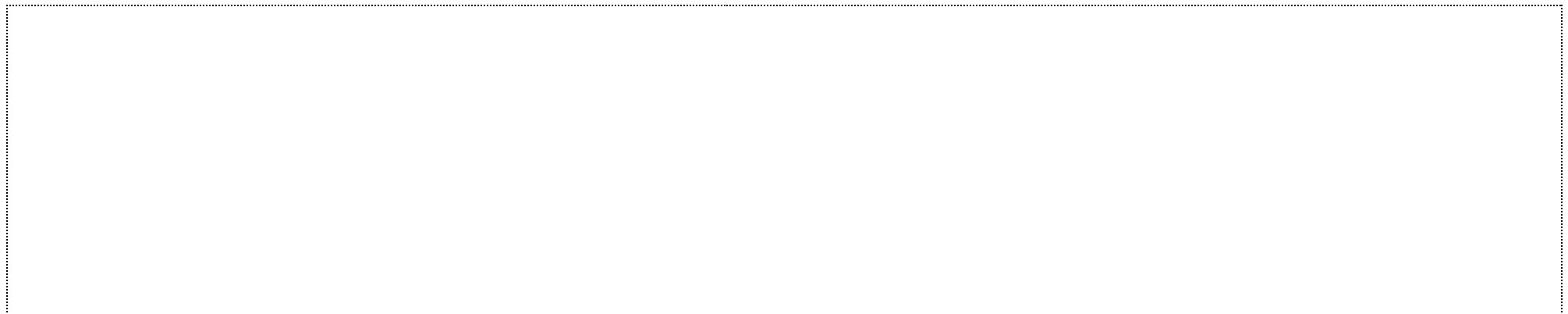
- Pozor na odborné výrazy, cizí slova, ...
- Spíše kratší věty

Informace jsou srozumitelné a přehledné



- Dělení obsahu do odstavců `<p>` s nadpisy
- Formuláře `<fieldset>`
- Rozsáhle výběry hodnot `<optgroup>`

Informace jsou srozumitelné a přehledné



- Odkazy „přejít na obsah“ a „přejít na navigaci“
- Důležitá sdělení vždy na začátku

Přeskakování navigace

```
<h1>Nadpis stránky</h1>
<a href="#content" class="hidden">Přejít na obsah</a>
<div class="menu">
    <h2>Menu</h2>
    ...
</div>
<div class="hlavni">
    <a name="content"></a>
    ...
</div>
```

Vlastnosti přístupného webu

- A. Obsah stránek je dostupný a čitelný
- B. Práci s webovou stránkou řídí uživatel
- C. Informace jsou srozumitelné a přehledné
- D. **Ovládání webu je jasné a pochopitelné**
- E. Kód je technicky způsobilý a strukturovaný
- F. Prohlášení o přístupnosti webových stránek

Ovládání webu je jasné a pochopitelné

- Navigační část (menu) je zřetelně oddělena od obsahu
- Vždy na stejném místě
- V samostatném bloku
- Vhodné označit nadpisem

Ovládání webu je jasné a pochopitelné

- Snadné určení aktuální pozice (pokud uživatel příjde např. z vyhledávače)
- Snadný pohyb všemi směry v navigační struktuře
- Přímý odkaz na hlavní stránku
 - Pokud jsou použity rámy tak ze všech rámů

Ovládání webu je jasné a pochopitelné



- Mapa webu – strukturovaný seznam odkazů na všechny stránky
- Rychlejší hledání konkrétní informace
- Odkaz na každé stránce na stejném místě
- Příklad mapy webu (hrad.cz)

Ovládání webu je jasné a pochopitelné

- Element **<title>**
- Název celého webu i konkrétní stránky
- Vystihuje smysl dané stránky

Ovládání webu je jasné a pochopitelné

- Používat prvky **<label>**

Ovládání webu je jasné a pochopitelné



- Obvykle zobrazení chyb před formulářem
- Správné hodnoty musí zůstat vyplněné (formulář se nevymaže)
- Musí být řečeno, jak chybu odstranit (např. správný formát data)

Ovládání webu je jasné a pochopitelné

- Srovnej:
 - Více informací o produktu zde
 - Více informací o produktu
- U souborů jiných typů je nutno uvést typ (např. PDF) a velikost
- Atribut **title** nebo text ve stejném odstavci či bloku, jako odkaz

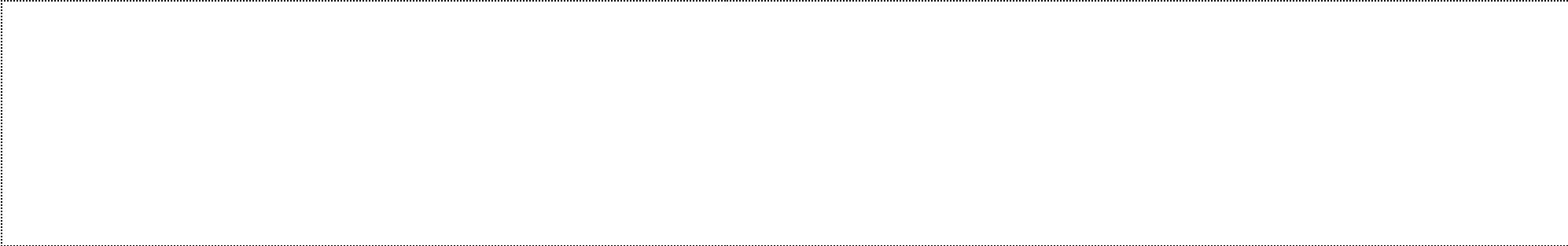
Ovládání webu je jasné a pochopitelné

- Některá zařízení prezentují rámy jednotlivě
- Je třeba vědět, co je obsahem každého rámu
- Atributy **name** a **title** značky **<frame>**

Vlastnosti přístupného webu

- A. Obsah stránek je dostupný a čitelný
- B. Práci s webovou stránkou řídí uživatel
- C. Informace jsou srozumitelné a přehledné
- D. Ovládání webu je jasné a pochopitelné
- E. Kód je technicky způsobilý a strukturovaný
- F. Prohlášení o přístupnosti webových stránek

Kód je technicky způsobilý a strukturovaný

- 
- Např. <address>, <h1> – <h6> apod.
 - Vždy používat pro daný účel
 - Nezneužívat pro žádný jiný účel (např. pro dosažení určitého vzhledu)

Kód je technicky způsobilý a strukturovaný

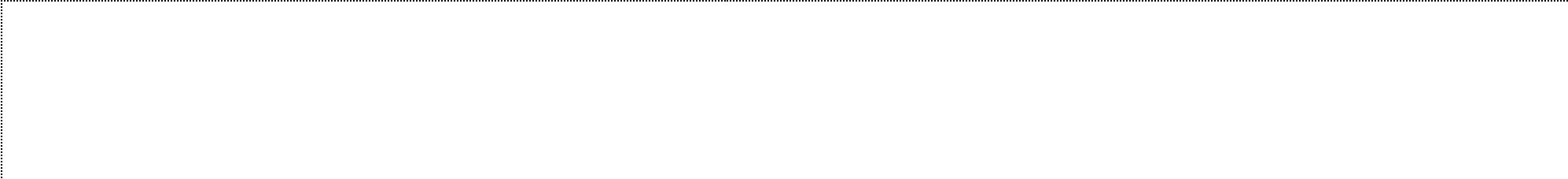


- Vychází ze specifikace jazyků HTML a XHTML
- V HTML jsou některé koncové značky nepovinné, takové se uvádět nemusí

Kód je technicky způsobilý a strukturovaný

- Atributy lang a xml:lang značky <html>
- V HTML pouze lang, v XHTML nejčastěji oba

Kód je technicky způsobilý a strukturovaný

- 
- Nezneužívat nadpisy a seznamy k něčemu jinému
 - Netvořit skutečné nadpisy a seznamy nějak jinak

Kód je technicky způsobilý a strukturovaný

- Značky <th>
- Naopak tabulky použité pouze pro rozvržení prvků na stránce nesmí mít záhlaví

Kód je technicky způsobilý a strukturovaný

- Na některých zařízeních nelze zobrazit tabulku, zobrazí (přečtou) se buňky za sebou

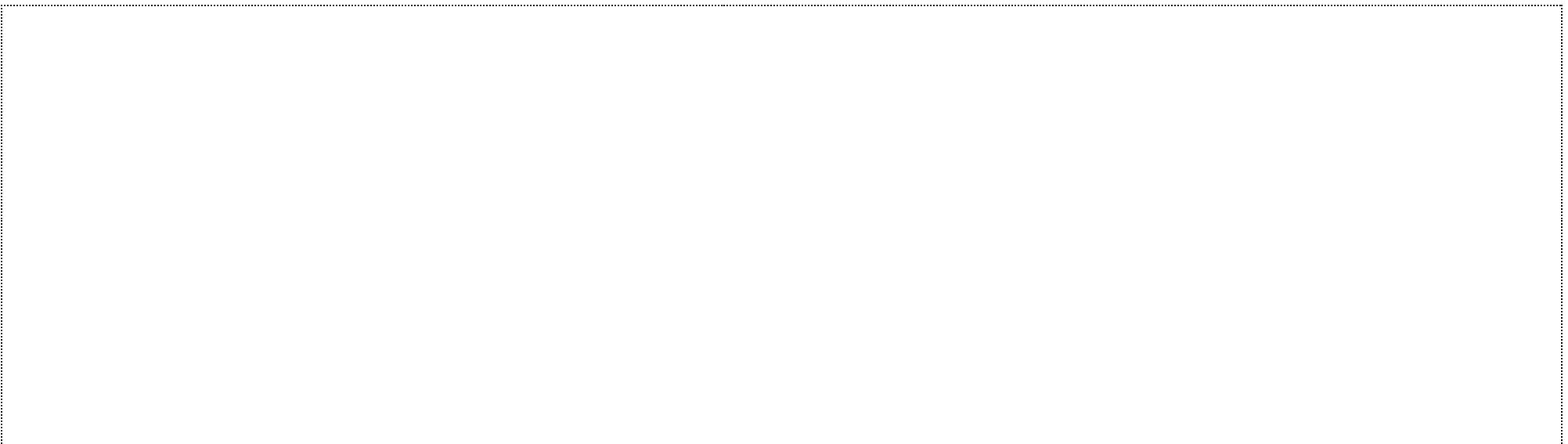
Vlastnosti přístupného webu

- A. Obsah stránek je dostupný a čitelný
- B. Práci s webovou stránkou řídí uživatel
- C. Informace jsou srozumitelné a přehledné
- D. Ovládání webu je jasné a pochopitelné
- E. Kód je technicky způsobilý a strukturovaný
- F. Prohlášení o přístupnosti webových stránek

Prohlášení o přístupnosti webových stránek

- Obvykle odkaz v zápatí apod.

Prohlášení o přístupnosti webových stránek



- **Technické překážky apod.**

Osvědčený postup

1. Web navrhujeme nejlépe odpočátku jako přístupný
2. Vytvoříme kostru (členění informací a navigace) bez definice vzhledu
 - Čisté (X)HTML pouze se strukturálními značkami a atributy
3. Přidáme pomocné odkazy (např. přeskočení navigace)
4. Vyzkoušíme použitelnost navigace nejlépe lynxem
5. Pomocí CSS definujeme vzhled a skryjeme pomocné odkazy
 - Pokud bylo nutno změnit HTML kód tak zpět na 4.
6. Vložíme vlastní obsah
7. Po celou dobu dbáme na dodržování zmíněných pravidel

Skrývání obsahu

- Skrývání úplné (části, které se právě nemají zobrazovat)
 - `display: none;`
 - `visibility: hidden;`
 - Obsah se nezobrazí, hlasová čtečka jeji nepřeče

Skrývání obsahu (II)

- Skrývání pouze na obrazovce (pomocné odkazy apod.)
 - Absolutní pozicování mimo obrazovku
 - Na obrazovce se nezobrazí, hlasová čtečka přečte

```
<a href="#content" class="invisible">Přeskočit navigaci</a>
.invisible {
    position: absolute;
    top: -500px;
    left: 0;
    width: 1px;
    height: 1px;
    overflow: hidden;
}
```

Přístupové klávesy

- Některé odkazy v dokumentu mohou být aktivovány klávesovou zkratkou
- Užitečné hlavně když nelze použít myš
- Zkratky jsou mapovány na čísla na klávesnici
- Aktivace se liší podle prohlížeče a OS:
 - Ve Windows obvykle levý Alt + Shift + číslo
- V česku se v posledních měsících prosazuje jednotný Standard klávesových zkratek

Zadání v HTML

- U prvků `<a>` případně `<input>`
- Atribut `accesskey`

```
<a href="http://www.tentoweb.cz/zkratky/"  
    accesskey="1" class="skryj">  
    Klávesové zkratky na tomto webu  
</a>
```

- Odkaz často bývá skrytý (aktivuje se pouze klávesnicí)

Obvyklý standard zkratek

- 0 Na obsah stránky - odkaz vedoucí na začátek unikátního obsahu aktuální stránky
- 1 Nápověda ke klávesovým zkratkám - otevře stránku s informací o použitých klávesových zkratkách
- 2 Hlavní strana - odkaz na úvodní stránku
- 3 Mapa stránek - nepovinné - pokud je to možné, odkáže na mapu stránek
- 4 Vyhledávání - nepovinné - pokud je to možné, přenese kurzor do textového políčka vyhledávacího formuláře
- 5 až 9 Volitelné funkce

Odkazy

- Přístupnost
 - <http://pristupnost.nawebu.cz/>
 - [Blind friendly web](#)
 - [Cynthia Says](#)
 - [Standard klávesových zkratek](#)