1. Struktura stránky, zásady při psaní kódu, MVC pattern

Web pro kodéry

(Petr Kosnar, ČVUT, FJFI, KFE, PINF 2008)

Obsah

- Terminologie
- Prezentace x Obsah
- Struktura kódu
- Sémantika kódu
- Struktura stránky
- Šablony
- Template engine
- MVC pattern
- Další zdroje informací

Terminologie

- **Doména** – jednoznačné jméno počítače nebo serveru na internetu. (např. www.cvut.cz)
  - Dělí se na několik řádů. Národní a mezinárodní koncovky jako .cz, .sk, .com, .org a další jsou tzv. TLD (top level domain), neboli domény prvního řádu. Domény ve tvaru cvut.cz je druhého řádu, fjfi.cvut.cz je třetího řádu, atd.
- **URL** - (uniform resource locator) je řetězec s určitou strukturou, který slouží k určení umístění zdrojů (dokument nebo služba) na internetu.
  - definuje doménovou adresu, umístění na serveru, protokol a parametry.
    (např. http://en.wikipedia.org:80/w/wiki.phtml?title=pork&action=edit)
- **Web** – ucelený soubor dokumentů (obvykle HTML stránek) dostupný pod určitou doménou nejnižšího řádu (např.všechny stránky a dokumenty dostupné pod adresou www.cvut.cz)
- **Stránka** – jedna konkrétní HTML stránka nebo dokument. Stránka je fragment webu.
  (např. http://www.cvut.cz/cs/ao/vedeni-cvut)

Prezentace × Obsah

- Každá webová stránka se dá rozdělit na dvě části – Prezentační a obsahovou

- Obsah
  - **Co** chce stránka sdělit
  - Text, tabulková data, grafy, fotografie, audio a video nahrávky...
- Prezentace
  - **Jakou formou** to chce stránka sdělit
  - Grafické prvky, CSS, fonty, barvy, obrázky bez obsahové důležitosti (ozdobné prvky), layout stránky (rozložení jednotlivých částí)...

Struktura kódu

Snaha o...

- Oddělení prezentace od obsahu
  - Vzhled definovaný jen pomocí CSS
  - Obsah jen v (X)HTML
- Maximální stručnost kódu, eliminace obsahově nedůležitých prvků využívaných jen pro prezentaci
- Sémanticky správný kód
- Navigace a funkční prvky stránky nezávislé na platformě, prohlížeči, uživatelkém nastavení
  - Funkční menu bez javascriptu, odpovídající alternativní texty k důležitým obrázkům...
- Plná použitelnost stránky bez prezentační části
  - s vypnutými styly, obrázky, javascriptem
- Snadno stylovatelný (X)HTML kód

Struktura kódu – prakticky

- Nepoužívat elementy <font>, <blink>, , <marquee>... A další, čistě prezentační, které nemají obsahový význam
- Nepoužívat atributy color, align, width, height... (do CSS)
- Pro tvorbu layoutu používat jako obalové prvky jednotlivých bloků prvek
- Nepoužívat
, případně   a pod. Pro účely odsazení
- Upřednostňovat strukturální prvky (ul, dl...)
- Elementům přiřazovat odpovídající identifikátory pomocí atributů class a id.
  - Identifikátory pojmenovávat podle významu či smyslu, ne podle vzhledu, který daný prvek má (žádné „levy-sloupec“, „zeleny-nadpis“, atp. Místo toho raději „hlavni-nadpis“, „primarni-menu“). V případě změny vzhledu či umístění elementu v layoutu stránky by byl název zavádějící.

Sémantika kódu

- Sémantika je nauka o významu slov, znaků a symbolů

- Sémantika kódu (například (X)HTML) popisuje význam jednotlivých tagů a jejich určení.
  - Např. <p> značí odstavec, <h1> nadpis nejvyšší urovně (a důležitosti).
- Sémanticky správný kód je založen na následujících zásadách:
  - Nepoužívat prezentační kód v (X)HTML (K tomu jsou určené kaskádové styly)
  - Používat významově vhodné tagy. Neřídit se jejich implicitním vzhledem, ale významem! (tedy například <cite> použít pro citace, i v případě, kdy nechceme, aby byl jejich text odsazený a v kurzívě – to lze upravit pomocí CSS)

Sémantika kódu - prakticky

- Každý tag má nějaké určení. <h1> je určený pro nadpisy první úrovně (nejdůležitější), pro odstavec, atp.
- Každý odstavec tedy uzavřít do prvků a místo souvislého toku textu v jednom bloku rozděleném pomocí
- Nadpisy uzavřené v <h1> a </h1> jsou významově mnohem důležitější, než například nadpis v <h4></h4>
- Nepoužívat prvky v situacích, kde se logicky a významově nehodí. Tedy například nepoužívat <h3> v situaci, kdy chci text „jen“ zvýraznit a není to žádný nadpis
- Pro případ zdůraznění používat , pro případ silnějšího zdůraznění používat . Elementy <i> a <b> jsou čistě prezentační, určují jen grafickou podobu (kurzívu, tučný text), ale nemají sémantický význam – Nepoužívat!
- Pro grafické úpravy bez obsahového významu použít například či s příslušnou class nebo id a s nastylováním v CSS
- Nepoužívat názvy tříd podle vzhledu, ale podle smyslu (např. místo class=„červený-sloupec “ použít class=„novinky“)

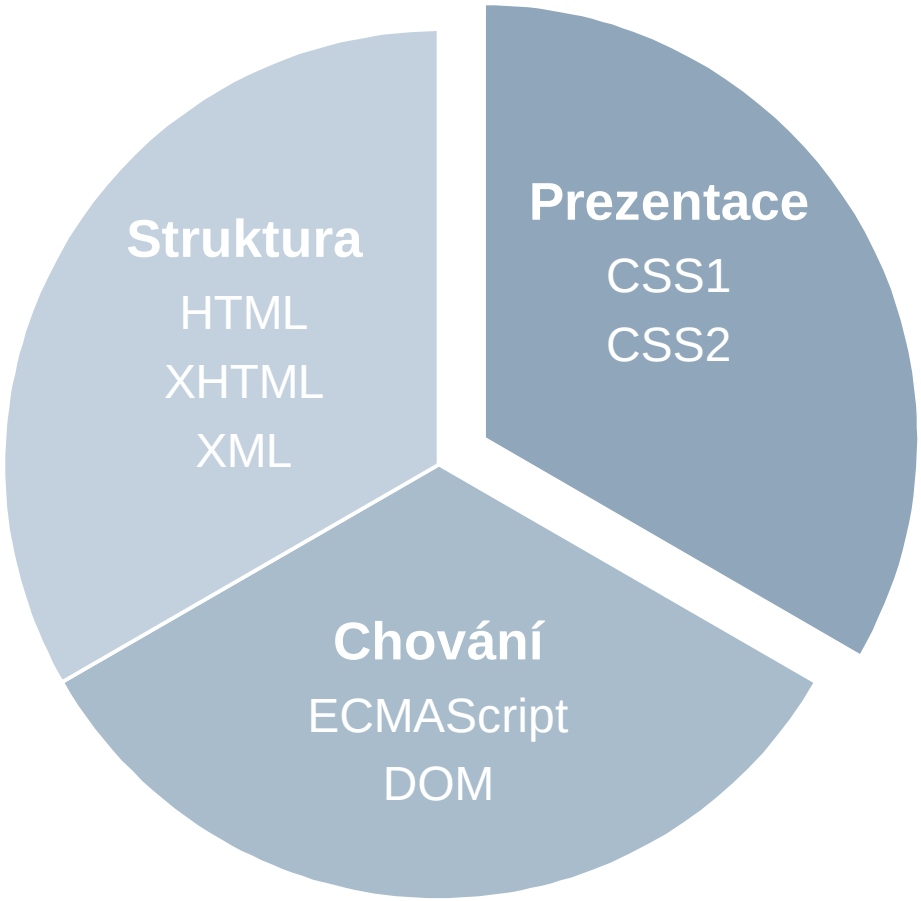
XHTML - hlavní sémantické elementy

- **em** - vyznačuje zvýraznění (emphasis).
- **strong** - označuje důraznější zvýraznení.
- **dfn** - obsahem tohoto elementu je pojem nebo definice (definition).
- **code** - používá se k označení nějakého zdrojového programového kódu.
- **samp** - vyznačuje vzorový výstup programů, skriptů, nějaký obecný vzorek
- **kbd** - indikuje text, který má být zadán uživatelem.
- **var** - označuje proměnnou, její vzorovou hodnotu apod.
- **cite** - označuje citovaný zdroj, odkaz na další zdroje nebo citaci.
- **abbr** - označuje zkratku (abbreviation), jejíž plné znění by se alespoň při prvním výskytu v dokumentu mělo nacházet v jeho atributu title. Příklady zkratek jsou XHTML, URI, ČR, nebo třeba ČVUT nebo FJFI;-)
- **acronym** - používá se k uzavírání zkratkových slov. Ty se narozdíl od zkratek vyslovují většinou jako jedno slovo, ne po jednotlivých písmenech. Příklady zkratkových slov jsou NATO, NASA nebo Čedok.
- **q** – slouží k uzavírání citací (řádkový, neboli inline element). Automaticky také doplňuje uvozovky okolo textu, který je v něm uzavřen
- **blockquote** - slouží k uzavírání citací (blokový element)

Struktura stránky

- Na každé stránce by mělo být přehledné hlavní navigační menu, pokud možno jednotného vzhledu a funkce napříč celým webem.
- Počítat s tím, že každá stránka může být první (nebo i jediná), kterou návštěvník uvidí. Nespoléhat na to, že všichni přijdou nejprve na homepage (návštěvy z vyhledávačů, přímých odkazů...)
  - Umožnit z každé stránky přístup na homepage, případně do nadřazené úrovně ve struktuře webu.
  - Informace o tom, na jakém webu se nacházíme na každé stránce (logo firemního webu, název portálu, atp.)
- Užitečná je mapa webu – stránka se strukturovaným přehledem všech stránek, které se na webu nachází
- Drobečková navigace (bread-crumb)

Svatá Trojice webových standardů

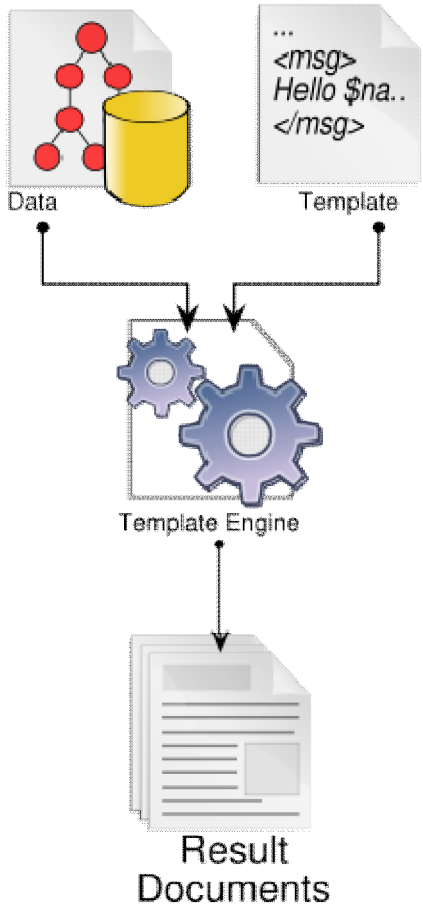


Šablony

- Řešení pro správu prezentačních aspektů webu tak, že jsou oddělené od logiky.
- Prezentační část kódu je oddělena od logiky webu a je vytvořena univerzální kostra, do které je vkládán obsah.
- Obsah je vkládán dynamicky pomocí nějakého šablonovacího enginu, nebo staticky (ručně)
- Šablona definuje layout, barvy, fonty a cekový vzhled stránky bez ohledu na vkládaná data
- Snadná změna celkového vzhledu stránek bez zásahu do jejich obsahu
- Možnost opakovaného použití šablony pro další projekty
- Oddělení prezentační a informační vrstvy dokumentu
- Umožňuje efektivní rozdělení práce v týmu (grafika a funkcionalita zvlášť)
- Šetří práci a čas – z jedné šablony je produkováno velké množství konečných dokumentů

Šablonovací engine

- Využívá skriptovací jazyk běžící na serveru (PHP, Ruby, Java, Python,...)
- Zajišťuje kombinaci obsahu a prezentace a generuje výsledný dokument předkládaný uživateli
- Obvykle zpracovává data z databáze nebo XML souboru
- Hlavní funkce:
  - Práce s proměnnýma a funkcemi
  - Náhrady a formátování textu
  - Vkládání souborů a fragmentů stránky
  - Podmíněné formátování a cykly
- Nejznámější:
  - Smarty (PHP), eRuby (Ruby), FreeMarker (Java), ASP.net (C#, VB.net)



Ukázka šablony (Smarty)

Šablona

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML
   1.1//EN"
   "http://www.w3.org/TR/xhtml11/DTD/xhtml1
   1.dtd">
<html>
<head>
   <title>{$title_text}</title>
   <meta http-equiv="content-type"
   content="text/html; charset=utf-8" />
</head>
<body>
   <p>Dnes je:
   {$smarty.now|date_format:"%d.%m.%Y"}</p>
   <p>{$body_text}</p>
   <p>{"text, který bude zkrácen ve
   SMARTY"|truncate:14:"..."}</p>
   {* výsledek: text, který bude... *}
</body
</html>
```

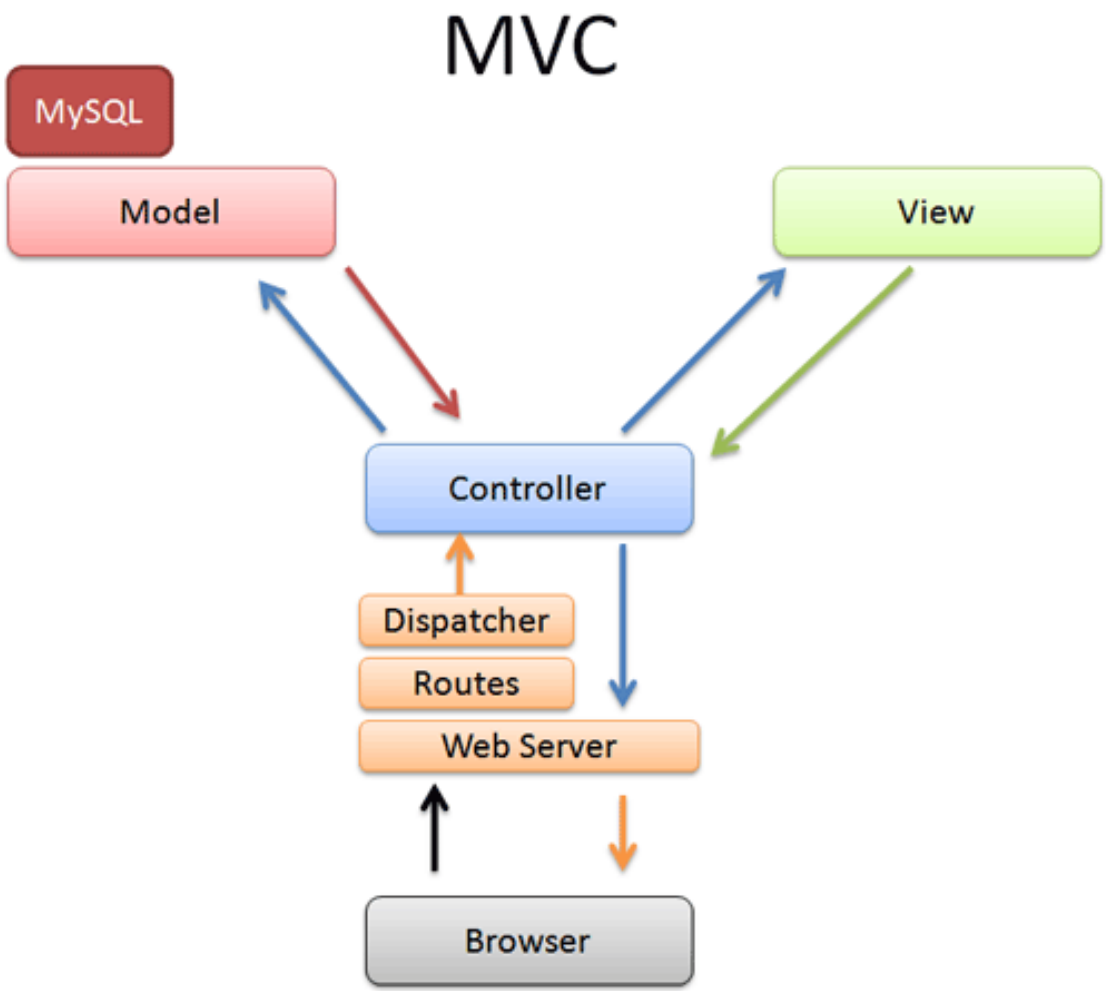
Výkonný PHP skript

```
require_once( "classes/Smarty.class.php" );

$smarty = new Smarty();
$smarty->assign('title_text', 'Toto je text
   titulku');
$smarty->assign('body_text', 'Toto je text
   těla stránky');

$smarty->display('index.tpl');
```

MVC Pattern aneb Model-View-Controller

- Softwarová architektura, která rozděluje datový model aplikace, uživatelské rozhraní a řídicí logiku do tří nezávislých komponent tak, že modifikace některé z nich má minimální vliv na ostatní.
- Základní komponenty:
  - **Model** – Výkonná logická funkce, práce s daty v databázích, výpočty, zpracovávání dat,...
  - **View** – Vykresluje konečný dokument (funkce šablony)
  - **Controller** – zajišťuje interaktivitu, reaguje na akce uživatele a rozhoduje o jejich zpracování, předává data zpracovaná a produkovaná Modelem do šablony View,
  - Propojuje logickou vrstvu (Model) s prezentační (View)
- MVC využívá mnoho PHP frameworků
  - Zend Framework, CakePHP, TinyMVC, Symfony, Nette, CodeIgniter...

MVC schema


MVC
MySQL
Model
View
Controller
Dispatcher
Routes
Web Server
Browser

Další informace

- http://interval.cz/serialy/xhtml-kompletni-pruvodce
- http://www.pixy.cz/dogma/dogmaw41/cs/index.html
- http://www.jaknaweb.com
- http://www.alvit.de/handbook
- http://www.w3schools.com
- http://interval.cz/clanky/tvorba-layoutu-webu-prakticky-prehled
- http://www.w3schools.com/tags/ref_byfunc.asp

Kontrolní úkol

- Zadaný textový dokument zpracovat do vhodně kódovaného a stylovaného (X)HTML

- Požadavky:
  - Validní HTML nebo XHTML
  - Validní CSS
  - Sémanticky správný kód
  - Oddělení obsahové a grafické části