Gymnázium a Jazyková škola s právem státní jazykové zkoušky Zlín
2017

# Tvorba webových stránek a kontaktního formuláře

závěrečná práce ze semináře z informatiky

**Kateřina Krhovská, Č4.B**

# Prohlášení

Prohlašuji, že jsem svou závěrečnou práci vypracovala samostatně a použila jsem pouze zdroje uvedené v přiloženém seznamu.

# Poděkování

Ráda bych touto cestou poděkovala panu učiteli Michalu Miklášovi za vstřícnost a cenné rady při tvorbě nejen této práce.

# Abstrakt

Tato práce se zabývá tvorbou responzivní webové stránky pro firmu Arisoft, formuláře a validací formuláře pomocí PHP.

# Abstract

This work deals with the creation responsive website for company Arisoft, form and form validation using PHP.

## Klíčová slova

Responsivní web, HTML, PHP, CSS

## Keywords

Responsive website, HTML, PHP, CSS

# Obsah

# 1 Úvod

## 1.1 Výběr práce

Vzhled webových stránek bylo na čase zmodernizovat. Jelikož původní verze nebyla responzivní, začala jsem úplně od začátku. Web totiž navštíví potencionální zákazníci často z mobilního zařízení, kteří hledají informace o službách, které firma nabízí, proto by měl web být přehledný a vhodný pro zobrazení na mobilu.

## 1.2 Cíl práce

Hlavním cílem této práce je vytvořit responzivní webovou stránku, která obsahuje kontaktní formulář.

# 2 Tvorba stránky

Jako první jsem si vytvořila nový soubor v programu Adobe Dreamviewer a do něj jsem si zkopírovala text z dokumentu Wordu, který jsem dostala spolu s dalšími požadavky, jak by měl web vypadat. Text jsem si rozvrhla do různých sekcí a následně jsem ho umístila mezi tagy[1] např. `<h1></h1>`. Pomocí tagů jsem si tedy vytvořila nadpisy, podnadpisy, odrážky, pododrážky a odstavce.

## 2.1 Menu stránky

Nad veškerý zdrojový kód jsem začala tvořit menu. Nejprve jsem si vytvořila div[2], který jsem si pojmenovala menu. Rozhodla jsem se, že celý web bude jedna stránka, jehož sekce budou pod sebou, proto nepíšu k sekci odkaz, ale přidávám kotvu k určitému divu. Poslední odrážka slouží k zobrazování ikonky třech čar nad sebou, když se zmenší rozměr webu, například prohlížení webu na tabletu či mobilu.

```
<ul class="menu" id="myTopnav">
<li><img src="logo.png" alt="Arisoft" height="30"></li>
<li><a class="ofirme" href="#ofirme">O firmě</a></li>
<li><a class="nabidkasluzeb" href="#nabidkasluzeb">Nabídka služeb</a></li>
<li><a class="nabidkasoftware" href="#nabidkasoftware">Nabídka software</a></li>
<li><a class="kontakt" href="#kontakt">Kontakt</a></li>
<li class="icon">
<div class="menuicon" onclick="myFunction(this)">
<div class="bar1"></div>
<div class="bar2"></div>
<div class="bar3"></div>
</div>
</li>
</ul>
```

Aby menu bylo responzivní, musela jsem použít JavaScript[3]. Jelikož v JavaScriptu neumím, vyhledala jsem si na internetu zdrojový kód pro tuto akci, který jsem našla na stránce:
`https://www.w3schools.com/howto/howto_js_topnav_responsive.asp`.

```
<script>
function myFunction(){
var x = document.getElementById("myTopnav");
if (x.className === "topnav"){
x.className += " responsive";
}else{
x.className = "topnav";
}
}
</script>
```

[1] Tag – značka
[2] Div – blokový element
[3] JavaScript – multiplatformí, objektově orientovaný skriptovací jazyk

Jedna z věcí, která mi dala nejvíce zabrat bylo právě responzivní menu. Pro efekt změnění čar pod sebou do písmena X jsem si musela kód podrobněji přečíst, pochopit a lehce ho upravit. Úpravy také následovaly v CSS[4].

```
<script>
function myFunction(x){
x.classList.toggle("change");
var y = document.getElementById("myTopnav");

if (y.className === "menu"){
y.className += " responsive";
}else{
y.className = "menu";
}
}
</script>
```

CSS pravidla, díky kterým dochází k transformaci čar:

```
.menuicon{
display: inline-block;
cursor: pointer;
padding: 5px;
}

.bar1, .bar2, .bar3{
width: 30px;
height: 4px;
background-color: white;
margin: 6px;
transition: 0.4s;
border-radius: 20px;
}

.change .bar1{
-webkit-transform: rotate(-45deg) translate(-6px, 6px);
transform: rotate(-45deg) translate(-6px, 6px);
}

.change .bar2{
opacity: 0;
}

.change .bar3{
-webkit-transform: rotate(45deg) translate(-8px, -8px);
transform: rotate(45deg) translate(-8px, -8px);
}
```

[4] Cascading Style Sheets – Kaskádové styly

## 2.2 Slideshow

Abych zvýšila atraktivitu webu, pro zobrazení novinek jsem zvolila slideshow. Slideshow je něco jako prezentace, zobrazuje několik po sobě jdoucích obrázků. Pro zobrazování následujících obrázků jsem volila automatické zobrazování po určitém uplynutí časevého intervalu, ale také manuální, kdy si uživatel sám klikne na šipku, buď předchozí či následující, nebo pod slideshow na puntík. Puntíky jsou zde pro větší přehled, které ukazují aktuálně zobrazený slide[5]. Aktuální slide má modrou barvu, ostatní jsou šedé.


Obrázek 1: Slideshow

Pro každý slide si vytvořím nový div se stejným názvem „mySlides fade“, který pak budu upravovat vzhled v CSS.

```
<div class="mySlides fade">
<img src="domovnik1.jpg" style="width:100%">
</div>
```

Dále je také potřeba vytvořit span[6] pro puntík, který bude určovat, jestli se slide právě zobrazuje nebo ne. `<span class="dot" onclick="currentSlide(1)"></span>`

K vytvoření šipek, po stranách slidů jsem vytvořila odkazy:

```
<a class="prev" onclick="plusSlides(-1)">&#10094;</a>
<a class="next" onclick="plusSlides(1)">&#10095;</a>
```

kde `onclick="plusSlides(-1)` se váže na funkci ve scriptu. Celkový script pro zobrazování slidů vypadá takhle:

```
<script>
var slideIndex = 1;
showSlides(slideIndex);

function plusSlides(n){
```

[5]Slide – jeden snímek (stránka) v prezentaci

[6]Span – HTML značka, tzv. řádkový element, tzn. nedělá za sebou zalomení řádku

```
showSlides(slideIndex += n);
}

function currentSlide(n){
showSlides(slideIndex = n);
}

function showSlides(n){
  var i;
  var slides = document.getElementsByClassName("mySlides");
  var dots = document.getElementsByClassName("dot");
  if (n==undefined){n = ++slideIndex}
  if (n > slides.length) {slideIndex = 1}
  if (n < 1) {slideIndex = slides.length}
  for (i = 0; i < slides.length; i++){
  slides[i].style.display = "none";
  }
  for (i = 0; i < dots.length; i++){
  dots[i].className = dots[i].className.replace(" active", "");
  }
  slides[slideIndex-1].style.display = "block";
  dots[slideIndex-1].className += " active";

  setTimeout(showSlides, 5000);
  }
</script>
```

Celý kód má zařídit, aby se po prvním zobrazení webové stránky zobrazil první slide a v časovém intervalu pěti sekund se změnil na druhý slide, jestliže uživatel už neklikl na odkaz šipky nebo na puntíky.

## 2.3 Favicon

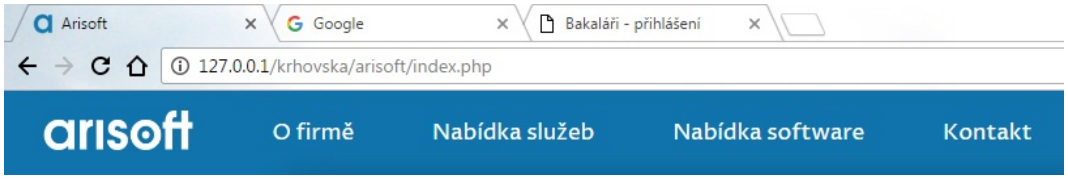
Favicon, přesněji `favicon.ico` je obrázek neboli ikonka o velikosti 16x16 px nejčastěji se zobrazující v adresním řádku nebo na panelu se stránkou. Při velkém počtu otevřených záložek je ikonka hlavním orientačním prvkem, jak požadovanou záložku najít. Z tohoto důvodu jde považovat absenci ikony za velkou chybu s ohledem na použitelnost.

Pro tvorbu faviconu jsem použila webovou stránku `http://www.xiconeditor.com/` která generuje favicony. Tahle webová aplikace nám umožní nakreslit si ikonu, nebo importovat obrázek, ze kterého udělá soubor s příponou .ico.

Do hlavičky, kde jsem mimo jiné napsala odkaz pro CSS a fonty použité pro web, jsem připsala následující:

```
<link rel="shortcut icon" type="image/x-icon" href="favicon.ico">
```

Na obrázku můžeme vidět favicony na různých webových stránkách a také když například web `http://ezk.gjszlin.cz/login.aspx` nemá definovaný favicon.




Obrázek 2: Ukázka faviconu

## 2.4 GoogleMaps

Aby stránka vypadala důvěryhodně, mělo by být uvedeno jméno zástupce firmy, ale také adresa sídla, aby bylo vidět, kde skutečně firma sídlí, že se nejedná jen o nějakou podvodnou firmu. Také je vhodné přidat mapu, aby bylo zřejmé jak se k nám může zákazník dostat. Jednoduchým a efektivním zobrazením mapy jsem dosáhla pouze zkopírování vygenerovaného kódu, po vyplnění polí na adrese: `https://embedgooglemaps.com/cs/`, který se již postará o zobrazení toho správného místa na Google mapách.

# 3 Tvorba kontaktního formuláře

V dnešní době by každá webová stránka firmy měla obsahovat kontaktní formulář. Formulář slouží ke komunikaci mezi firmou a případným zákazníkem, proto je důležité, aby formulář byl přehledný a jasný. Při stisknutí na tlačítko „Odeslat“ se zkontrolují zadané údaje. Když je údaj zapsaný špatně, napíše se uživateli, kde udělal chybu. Když je formulář vyplněný korektně, vypíšou se nám zadané údaje.


Obrázek 3: Ukázka sekce „Kontakt“ pro mobilní verzi – kontaktní formulář

## 3.1 Rozvržení formuláře

Když jsem se rozhodla pro tvoření formuláře, jako první jsem hledala na internetu inspiraci, poté jsem se rozhodla navrhnout si formulář na papír. Neměl to být složitý a obsáhlý formulář, stačilo jen vytvořit textové pole pro doplnění jména, příjmení, e-mailu, možno také telefonu a to nejdůležitější – komentář.

Formulář se nachází v sekci „Kontakt“, kde mimo formuláře jsou i informace o firmě a kontakt na ni. Celkový formulář píšu mezi tagy `<form></form>`. K tagu `<form>` připíšu atributy jako je `class` (třída, pomocí které pak můžu formulář ostylovat v CSS) `action` (adresa skriptu, který zpracuje informace z formuláře) `method` (metoda, jakým způsobem se budou přenášet informace z formuláře na server). Atribut `method` nabývá hodnot `POST` nebo `GET`. Když atribut nenapíšu, použije se metoda `GET`, která posílá data jako součást `URL`. Data jsou vidět v adresním řádku (uživatel je vidí). Proto by se měla metoda `GET` používat pro data krátká a pro data

u kterých nevadí, že si je uživatel přečte, případně je v adresním řádku změní.

```
<form class="form" action="sent_email.php" method="post">
```

Vytvoření textového pole pro každý doplňující údaj vypadal vždy podobně. K tagu `<label>` jsem napsala atribut `for=""` a mezi uvozovky dopsala, k čemu bude textové pole určené. Abych měla na jednom řádku text k textovému poli, použila jsem tag `<span>`, který nedělá za sebou zalomení řádku, narozdíl od tagu `<div>`. Dále za `<span>` následuje tag `<input>`. Je to tzv. vstupní a nepárový tag, proto ho neukončujeme `</input>`, ale pouze `>`. Input v sobě zahrnuje celou škálu různých kolonek, tlačítek a přepínačů, to všechno závisí na atributu type. Pro normální zapisování textu do textového pole jsem zvolila `type="text"`. Dále jsem dopsala atributy jako jsou `name`, `id` a `required`. Atribut `required` znamená, že prohlížeč zabrání odeslání, pokud pole nebude vyplněno.

```
<label for="jmeno">
<span>Jméno:</span><input type="text" name="jmeno" id="jmeno" required> *
</label>
```

Textové pole pro vyplnění příjmení je velmi podobné, ale v poli pro vyplnění e-mailu se zobrazuje něco zajímavého.


Obrázek 4: Ukázka atributu placeholder

Atribut `placeholder` je šedý text uvnitř textového pole, který při prvním kliknutí zmizí. Používá se zejména jako nápověda, aby uživatel věděl jak pole vyplnit, tudíž po začátku psaní text zmizí.

```
<label for="email">
<span>E-mail:</span><input type="text" name="email" id="email"
placeholder="jan.novak@seznam.cz" required> *
</label>
```

Další součástí formuláře je textové pole pro komentář. V předchozích příkladech jsem použila `input`, nyní použiju `textarea`[7]. Narozdíl od inputu je to párový tag. Původně jsem určovala šířku pole ve znacích pomocí atributu `cols` (např. cols=”30”) a výšku pole v řádcích pomocí atributu `rows` (např. rows=”5”), ale nakonec jsem vše definovala v CSS.

```
<label for="komentar">
<span>Komentář:</span><textarea name="komentar" type="textarea" id="komentar"
required ></textarea> *
</label>
```

Zde je zdojový kód v CSS, kde ho podrobněji rozeberu:

[7] Textarea – rozsáhlé vstupní pole, zobrazuje rámeček s lištou

```
.form textarea{
padding: 0px;
outline: none;
border: none;
border-bottom: 1px dashed #B1C2D8;
width: 275px;
height:40px;
overflow: hidden;
font-style: italic;
}
```

Pro `textarea` tedy platí, že nemá žádný obrys okolo rámečku, to nám definuje `outline: none`. Co se mi na textovém poli líbí, je světlá přerušovaná čára, která se mění na tmavou, pokud jsme kurzorem klikli do textového pole. To zařídíme pomocí selektoru `focus`:

```
.form textarea:focus,
.form input:focus{
border-bottom: 1px dashed #1073AB;
}
```

Dále jsem jen nadefinovala výšku a šířku, která je stejná jak ostatní textové pole a nastavila jsem, aby doplněný text byl psán kurzívou `font-style: italic`.

Nakonec jsem vytvořila tlačítko „Odeslat“, bez kterého by se žádný kontaktní formulář neobešel.

```
<label>
<span> </span><input type="submit" id="submit" value="Odeslat" />
</label>
```

Aby tlačítko dobře vypadalo, ostylovala jsem ho náslovně:

```
.form input[type=submit]{
background-color: #1073AB;
border: none;
padding: 8px 10px 8px 10px;
border-radius: 5px;
color: #B1C2D8;
font-style:normal;
box-shadow: 2px 2px 5px grey;
}
```

Při najetí kurzorem na tlačítko odeslat se změní barva a stín zmizí, aby udělal iluzi stisknutého tlačítka.

```
.form input[type=submit]:hover{
background-color: #0B527A;
box-shadow: none;
}
```

# 4 PHP, ověření zadaných údajů

Dále bych se chtěla zaměřit na ověření zadaných údajů. Vytvořila jsem si nový `php`[8] soubor, který jsem pojmenovala `sent_email.php` a připojila jsem ho atributem `action` v tagu `form`.

```
<form class="form" action="sent_email.php" method="post">
```

## 4.1 Kontrola jména a příjmení

Chtěla jsem zařídit, aby nešel odeslat formulář, když by bylo v poli pro jméno nebo příjmení méně než dva znaky a také, kdyby byly v poli napsané čísla či jiné znaky kromě písmen. Proto jsem si vytvořila proměnnou `$povoleneznaky`. Slabinou jsou tady znaky české abecedy. Přestože jsem hledala na internetu různé řešení, stále ověřování ignorovalo české písmena. Aby jméno nebo přijmení nemělo méně než dvě písmena, s tím nám pomáhá funkce `strlen`, která spočítá délku řetězce a vrátí nám počet znaků v řetězci.

```
$povoleneznaky = "/^[A-Za-z .'-]+$/";
if(!preg_match($povoleneznaky,$jmeno)){
  $error = $error."<p>Prosím, napište znovu vaše jméno.</p>";
}

if(strlen($jmeno)<2){
  $error = $error."<p>Prosím, napište znovu. Vaše jméno musí mít minimálně dvě
písmena.</p>";
}
```

## 4.2 Kontrola e-mailu a telefonu

Pro e-mail sepsat povolené znaky bylo o něco těžší, proto jsem si vyhledala na internetu, co takový e-mail může a nemůže obsahovat.

```
$povoleneznakyemail = '/^[A-Za-z0-9._%-]+@[A-Za-z0-9.-]+\.[A-Za-z]{2,4}$/';
if (!preg_match($povoleneznakyemail, $email)){
  $error = $error."<p>Zadaný e-mail je napsaný ve špatném formátu.</p>";
}
```

Vyplnění textového pole pro telefon není povinné. Jestliže se uživatel rozhodne kolonku vyplnit, musí telefon vyplnit ve správném tvaru. Pole totiž musí obsahovat pouze číslice, což zařizuje funkce `is_numeric` a také nesmí obsahovat více než devět číslic, to znamená, že v našem případě tam uživatel musí zadat pouze číslo bez telefonní předvolby. Tohle by rozhodně stálo za vylepšení o přidání možné telefonní předvolby.

```
$i = 0;
for($i=0;$i<strlen($telefon);$i++){
if(!is_numeric($telefon[$i])){
$error = $error."<p>Prosím, napište znovu. Váš telefon musí obsahovat pouze 9
```

[8] PHP - skriptovací programovací jazyk, určený především pro programování dynamických internetových stránek

```
číslic bez mezer.</p>";
}
}

if(strlen($telefon)<>9){
  $error = $error."<p>Prosím, napište znovu. Váš telefon musí obsahovat pouze 9
číslic bez mezer.</p>";
}
```

# 5 Ochrana e-mailu před spamem

Použila jsem techniku pro ochranu emailových adres při zachování „klikatelnosti“ – převod znaků emailové adresy do ASCII sekvencí. Všechny používané internetové prohlížeče po načtení automaticky ASCII sekvence nahrazují za normální znaky, takže uživatel i Outlook se již dostanou k čitelnému textu a ve zdrojovém kódu HTML přitom žádné použitelné emailové adresy nebudou. Princip je tedy takový, že každý znak, můžeme v HTML zapsat alternativní cestou a to přes jeho kód v ASCII. Například znak „A“ můžete napsat jako „65;“. To znamená, že tento kód:

```
<a href="mailto:&#105;&#110;&#102;&#111;&#64;&#97;&#114;&#105;&#115;&#111;
&#102;&#116;&#46;&#99;&#122;">
&#105;&#110;&#102;&#111;&#64;&#97;&#114;&#105;&#115;&#111;&#102;&#116;&#46;
&#99;&#122;</a>
```

uvidíme jako:

```
<a href="mailto:info@arisoft.cz">info@arisoft.cz</a>
```

Odkaz na ASCII tabulku, která ke každému znaku uvádí HTML hodnotu: `http://www.lookuptables.com/`

# 6 Responzivita

Web by měl být responzivní, aby se přizpůsobil každému displeji. Ať už si uživatel bude prohlížet web na mobilu, tabletu, počítači nebo jiném zařízení, bude pro něho web vždy přehledný a jednoduchý na obsluhu. Nemusí se tedy pro mobilní telefony vytvářet speciální mobilní verze webu. Stačí udělat jeden web na vše a měnit ho pouze v CSS pomocí `@media screen`. Nejen přehlednost je výhodou. Také společnost Google začala ve vyhledávání penalizovat weby, které nejsou přizpůsobeny mobilním zařízením, tím pádem responzivní web je snadněji vyhledatejnější. Pro zobrazení na mobilním zařízení je nutné vložit do hlavičky stránky `<meta>` značku `viewport`, která mobilnímu prohlížeči řekne, že se stránka chce přizpůsobovat skutečné šířce.

```
<meta name="viewport" content="width=device-width">
```

## 6.1 Media query

Nadefinuju media query pro rozlišení maximálně 850 px, protože kdyby se odrážky menu nevlezli vedle sebe, byly by pod sebou a to nechci.

```
@media screen and (max-width:850px){
ul.menu li:not(:first-child) {display: none;}
ul.menu li.icon{
  float: right;
  display: inline-block;
  padding-top:10px;
}
ul.menu.responsive{position: relative;}

ul.menu.responsive li.icon{
  position: absolute;
  right: 0;
  top: 0;
}
ul.menu.responsive li{
  float: none;
  display: inline;
}
ul.menu.responsive li a{
  display: block;
  text-align: left;
}
#box{
  width:95%;
  padding:0px;
  box-shadow: none;
}

#slideshow{display:none;}
```

V předchozím kroku jsem tedy zařídila responzivní menu, kde `ul.menu li:not(:first-child) {display: none;}` nám zařizuje, aby se odkazy na sekce schovaly a objevily se jen tři vodorovné čáry nad sebou - ikona pro menu. Dále jsem zúžila okraje pro hlavní div, pojmenovaný box, ve kterém je veškerý obsah stránky a také odstranila stín boxu a div, který obsahuje slideshow.

# 7 Krátce o zpracování textu LaTeXem

Pro zpracování tohoto dokumentu jsem zvolil sazbu systémem LaTeX, jelikož nabízí mnoho možností pro psaní nejen matematických textů, ale například i textů právě do informatiky. Dokument v LaTeXu se píše jako text, do kterého se ručně vpisují formátovací příkazy, podobně jako například v HTML, následně je nutné dokument přeložit. To na jednu stranu přináší nutnost pamatovat si příkazy nebo je vyhledávat, na druhou stranu je tak důsledně možné při tvorbě dokumentu nezávisle na sobě pracovat na obsahu dokumentu a na jeho vzhledu. Pokud například autor chce začít novou kapitolu, použije příslušný příkaz a nestará se vůbec o to, jakým a jak vysokým fontem bude nadpis kapitoly vysázen. Velikosti a typy fontů jsou dány stylem dokumentu a pouhou změnou definice stylu (jeden příkaz v záhlaví dokumentu) je možné zcela změnit vzhled dokumentu.

## 7.1 Šablona

Jsem ráda, že jsem mohla použít práci Jana Macáka jako šablonu, bez které by bylo pro mě o dost obtížnější dokumentaci správně vytvořit.

# 8 Použité programy

**Adobe Dreamweaver CS5** Adobe Dreamweaver je nejpoužívanější program pro tvorbu webových stránek. Software podporuje HTML, HTML5, XHTML, CSS a další. Nabízí všechny nástroje a možnosti potřebné pro vytvoření profesionálního webu.

**Adobe Photoshop CS5** Adobe Photoshop je grafický rastrový editor s velkým množstvím funkcí. Program jsem využila při práci s obrázky.

**Adobe Illustrator CS5** Adobe Illustrator je grafický vektorový editor, který jsem použila při tvorbě loga.

**WampServer64** WampServer je program pro tvorbu webových aplikací pomocí Apache, PHP a MySQL databáze. Program je velmi jednoduchý na ovládání a zaručuje bezproblémový chod a test webových aplikací.

**Google Chrome, verze 56.0.2924.87** Google Chrome je webový prohlížeč od společnosti Google, používala jsem ho k náhledu webové stránky a k vyhledávání informací.

**TEXnicCenter, 2.2 Stable** TEXnicCenter je editor sloužící pro sazbu textu v LATEXu. Pro usnadnění práce je zde mnoho funkcí jako vkládání symbolů nebo obrázků. Při psaní tohoto textu jsem využil distribuci MikTEX.

# 9 Závěr

Řekla bych, že vytvořený web by se mohl použít v praxi, kdyby se doladily některé detaily. Jsem ráda, že jsem si více méně sama poradila s problémy, které nastaly při tvorbě této stránky, případné nejasnosti se daly vyhledat na internetu. Také mám radost, že jsem mohla tátovi pomoct a ušetřit čas s modernizací jeho stránky.

# 10 Internetové zdroje

# Reference


[1] https://www.tvorba-webu.cz/php/formulare.php

[2] https://www.jakpsatweb.cz/html/formulare.html

[3] http://php.vrana.cz/http-metody-get-a-post.php

[4] http://www.xiconeditor.com/

[5] https://embedgooglemaps.com/cs/

[6] http://programujte.com/clanek/2006113003-ochrana-emailove-adresy-pred-roboty/