

Osnove izrade PHP aplikacija

Provjera obrazaca i slanje e-pošte

Izradio: Davor Cihlar

Plan

- ▶ Što ćemo naučiti
 - ▶ Automatsku provjeru obrazaca
 - ▶ Slanje e-pošte

Osnove izrade PHP aplikacija

Provjera obrazaca

Izradio: Davor Cihlar

Provjera obrazaca

- ▶ Želimo što manje razmišljati o ispravnosti podataka
 - ▶ Podatke iz obrasca želimo odmah koristiti, bez provjere na svakom koraku
- ▶ Osnovna ideja je zadati kako podaci u obrascu moraju izgledati i tek kad su svi ispravno uneseni da se pozove naša skripta
 - ▶ Dio sa samom provjerom i objavom grešaka želimo imati automatiziran

SmartyValidate

- ▶ SmartyValidate je proširenje za Smarty koje omogućuje provjeru obrazaca
- ▶ U predlošku se definiraju mjesta za prikaz greške
- ▶ U skripti se definiraju načini provjere
 - ▶ Ako je provjera uspješna, nastavlja se izvođenje skripte s provjerenim podacima
 - ▶ Ako provjera nije uspješna, prikazuje se ista stranica
 - ▶ Moguće je ispuniti obrazac s podacima koje je korisnik prije unio (ali krivo)
- ▶ **Više na:**
 - ▶ <http://www.phpinsider.com/php/code/SmartyValidate/>

Instalacija

- ▶ Potrebna je samo za one koji rade lokalno
- ▶ Skinite arhivu s internetske stranice SmartyValidate
- ▶ Sadržaj direktorija "libs" iz arhive kopirajte u "smarty3" direktorij
- ▶ Direktorij "plugins" kopirajte pokraj direktorija "smarty3"

Kostur skripte

```
session_start();
require_once('smarty3/SmartyValidate.class.php');

if (empty($_POST)) {
    SmartyValidate::connect($smarty, true);
    SmartyValidate::register_validator(...);
    ...
    $smarty->display('obrazac.tpl');
} else {
    SmartyValidate::connect($smarty);
    if (SmartyValidate::is_valid($_POST)) {
        SmartyValidate::disconnect();
        $smarty->display('uspjeh.tpl');
    } else {
        $smarty->assign('obrazac', $_POST);
        $smarty->display('obrazac.tpl');
    }
}
}
```

Povezivanje SmartyValidate + Smarty

- ▶ SmartyValidate povezuje se sa Smartyjem pomoću metode `connect`
 - ▶ Prvi parametar te metode je instanca klase `Smarty`
 - ▶ Drugi parametar je pretpostavljeno `false`, ali ako je postavljen na `true`, interne informacije o obrascu će biti resetirane
- ▶ Primjer prilikom prikaza obrasca:
 - ▶ `SmartyValidate::connect($smarty, true);`
 - ▶ Potrebno je resetirati interne informacije
- ▶ Primjer prilikom zaprimanja obrasca:
 - ▶ `SmartyValidate::connect($smarty);`

Registracija tipa provjere (1)

- ▶ Metodom `register_validator` određuju se tipovi provjere za svaki element kojeg želimo provjeriti
 - ▶ `SmartyValidate::register_validator('idProvjere', 'imeElementa', 'tipProvjere');`
- ▶ `idProvjere` označava identifikator provjere
- ▶ `imeElementa` označava ime elementa
 - ▶ Npr. iz `$_GET` ili `$_POST`
- ▶ `tipProvjere` označava naziv provjere
 - ▶ Postoji nekoliko ugrađenih tipova provjere

Registracija tipa provjere (2)

- ▶ Tipove provjera potrebno je registrirati samo prilikom prvog prikazivanja obrasca
- ▶ Informacije o provjerama spremaju se u sjednicu!
 - ▶ Uvijek je potreban `session_start`!

Tipovi provjere

- ▶ **Bitni ugrađeni tipovi provjere su:**
 - ▶ `notEmpty` - da li je element ispunjen
 - ▶ `isInt` - da li je element cijeli broj
 - ▶ `isNumber` - da li je element broj
 - ▶ `isEmail` - da li je element ispravna adresa e-pošte
 - ▶ `isURL` - da li je element ispravan URL
 - ▶ `isEqual` - da li su dva elementa s istim sadržajem
 - ▶ Korisno za provjeru lozinke
 - ▶ `isRange` - da li je element unutar zadanog raspona
 - ▶ `isLength` - da li element sadrži tekst određene duljine
 - ▶ `isFileType` - da li je poslana datoteka pravoga tipa

Parametriranje tipova provjere (1)

- ▶ Neki tipovi provjere zahtijevaju parametre
- ▶ Parametri se zadaju unutar `imeElementa`
 - ▶ Odvajaju se znakom dvotočke
 - ▶ Primjerice: `broj:1:5`
 - ▶ `broj` je ime elementa
 - ▶ `1` i `5` su parametri

Parametriranje tipova provjere (2)

- ▶ Bitni ugrađeni tipovi provjere koji zahtijevaju parametre
 - ▶ `isEqual` – ime drugog elementa za usporedbu
 - ▶ `register_validator('p', 'pass:pass2', 'isEqual')`
 - ▶ `isRange` – minimalni i maksimalni dopušteni broj
 - ▶ `isLength` – minimalna i maksimalna duljina
 - ▶ -1 ako nema ograničenja min. ili maks. duljine
 - ▶ `isFileType` – ekstenzije odvojene zarezom
 - ▶ `register_validator('d', 'datoteka:jpg,png', 'isFileType')`

Uspješnost provjere obrasca

- ▶ Valjanost podataka koje je korisnik unio možemo provjeriti metodom `is_valid`
- ▶ Parametar metodi je `$_GET` ili `$_POST`
 - ▶ Ovisi o tipu (metodi) obrasca

▶ Primjer:

```
▶ if (SmartyValidate::is_valid($_POST)) {  
    # podaci su dobro uneseni  
    SmartyValidate::disconnect();  
    $smarty->display('uspjeh.tpl');  
} else {  
    # neki ili vise podataka je krivo uneseno  
    $smarty->assign('obrazac', $_POST);  
    $smarty->display('obrazac.tpl');  
}
```

Izrada predloška

- ▶ Poseban Smarty element `validate` je uveden za prikaz greške
 - ▶ `validate` ima parametre:
 - ▶ `id` - `idProvjere`
 - ▶ `message` - poruka o grešci
 - ▶ Ukoliko ne prođe provjera vezana uz identifikator `idProvjere`, ispisuje se `message` na mjestu Smarty elementa
- ▶ Primjer:
 - ▶

```
{validate id="idProvjere" message="Poruka o gresci"}
```

Napredna dojava grešaka

- ▶ Moguće je dojavu greške spremiti u neku varijablu kako bi je kasnije prikazali ili provjerili da li je došlo do te greške

- ▶ Primjer:

```
{validate id="idProvjere"  
         message="Poruka o gresci"  
         assign="varGreska" }  
  
{if $varGreska}  
  Pazi na ovo: {$varGreska}  
{/if}
```

Ispuna obrasca sa starim podacima

- ▶ Ako korisnik unese neki podatak krivo, ne želi sve podatke ispočetka unositi
 - ▶ Frustrirajuće za veće obrasce!
- ▶ U primjeru za "Uspješnost provjere obrasca" prikazano je postavljanje cijele `$_POST` varijable
 - ▶ Tim podacima moguće je ispuniti obrazac!
- ▶ Primjer:
 - ▶

```
<input type="text" name="imePolja" value="{ $obrazac.imePolja|escape} ">
```

Zadatak (1)



- ▶ Napravite obrazac za registraciju korisnika pomoću Smartyja i SmartiValidatea
- ▶ Iskoristite kostur skripte prikazan ranije
- ▶ Obrazac za registraciju mora imati sljedeće elemente (polja za unos):
 - ▶ Ime i prezime (`notEmpty`)
 - ▶ Zaporka i ponovljena zaporka (`isEqual` i `isLength`)
 - ▶ Minimalna duljina zaporke: 4
 - ▶ Maksimalna duljina: neograničena (-1)
 - ▶ Dob (`isRange 12..120`)
 - ▶ Adresa e-pošte (`isEmail`)
- ▶ Nije potrebno implementirati proces registracije!

Osnove izrade PHP aplikacija Slanje e-pošte

Izradio: Davor Cihlar

Slanje e-pošte

- ▶ Internetska stranica ponekad treba moći poslati e-poruku korisniku
 - ▶ Provjera adrese e-pošte
 - ▶ Slanje nove lozinke
 - ▶ npr. korisnik je zaboravio staru i zatražio novu
 - ▶ Slanje pozivnice za registraciju na sustav
 - ▶ Obavijesti (moguće ih je slati samo u trenutku objave)
 - ▶ ...

mail()

- ▶ e-poruke je moguće slati jednostavnim pozivom na funkciju `mail`
- ▶ Primjer:
 - ▶

```
mail('ime.prezime@fer.hr',  
    'Naslov',  
    'Sadržaj poruke.');
```
- ▶ Više na:
 - ▶ <http://hr.php.net/manual/en/function.mail.php>
- ▶ Napomene:
 - ▶ Na Linuxu je prvo potrebno instalirati i konfigurirati MTA, npr. `exim4`
 - ▶ Na Windowsima je dovoljno konfigurirati SMTP u datoteci `php.ini`

Bogate e-poruke

- ▶ `mail` funkcija je vrlo jednostavna za koristiti
 - ▶ Ali kako priložiti datoteku u e-poruku?
 - ▶ Kako će na kraju izgledati hrvatski znakovi?
 - ▶ Kako poslati HTML poruku?
 - ▶ Sve od toga je moguće, ali uz dosta posla!
- ▶ U pomoć nam dolazi `Mail_mime` biblioteka iz PEAR (*PHP Extension and Application Repository*)
 - ▶ Ta biblioteka radi u suradnji s bibliotekom `Mail` koja služi samo za slanje e-poruka
 - ▶ `Mail` podržava više načina slanja e-poruka
 - ▶ Način slanja kojeg ćemo koristiti koristi upravo PHP-ovu `mail` funkciju

Biblioteka Mail

- ▶ Mail je sučelje za slanje e-poruka
- ▶ Podržava više načina slanja:
 - ▶ `mail` - preko PHP-ove funkcije `mail`
 - ▶ `sendmail` - preko `sendmail` programa
 - ▶ `smtp` - direktno preko SMTP poslužitelja
- ▶ Pozivom na statičku metodu `factory` (način slanja je parametar) instancira se nova klasa `Mail`
- ▶ Poruka se šalje pozivom na metodu `send` dobivene klase
 - ▶ Parametri su: određena adresa (ili polje s adresama), zaglavlja i tijelo poruke
- ▶ Više na:
 - ▶ <http://pear.php.net/package/Mail/docs>

Mail – primjer

```
▶ require_once 'Mail.php';
```

```
$text = 'Sadržaj poruke';
```

```
$hdrs = array('Subject' => 'Testna poruka');
```

```
$mail = Mail::factory('mail');
```

```
$mail->send('ime.prezime@fer.hr',  
          $hdrs,  
          $text);
```

MIME poruke

- ▶ MIME (*Multipurpose Internet Mail Extensions*) omogućuje:
 - ▶ Podršku za tekst koji nije samo ASCII
 - ▶ Binarne priloge
 - ▶ Višestruka tijela poruke
 - ▶ Zaglavlja koja nisu samo ASCII
- ▶ Zahvaljujući MIME-u moguće je slati HTML poruke
- ▶ Moguće je i u HTML poruke ugrađivati slike
 - ▶ Engl. *inline images*

Biblioteka Mail_mime

- ▶ Mail_mime omogućuje stvaranje zaglavlja i tijela poruke koja su MIME kodirana
- ▶ Stvorena zaglavlja i tijelo poruke direktno je moguće poslati
- ▶ Mail_mime je potrebno instancirati
 - ▶ Parametar konstruktora je oznaka za prelazak u novi red
 - ▶ "\r\n" ili "\n"
 - ▶ Zato što se koristi biblioteka Mail, taj parametar mora biti "\n"
- ▶ Više na:
 - ▶ http://pear.php.net/package/Mail_Mime/docs

Mail_mime – kostur

```
▶ require_once 'Mail.php';
  require_once 'Mail/mime.php';

$headers = array('Subject' => 'Naslov');

$mime = new Mail_mime("\n");

// postavljajanje sadržaja...

// obavezan redoslijed!!!
$body = $mime->get(); // sastavi tijelo poruke
$headers = $mime->headers($headers); // sastavi zagl.

$mail = Mail::factory('mail');
$mail->send('ime.prezime@fer.hr',
           $headers, $body);
```

Slanje tekstualnih poruka

- ▶ Metodom `setTXTBody` postavlja se čisti tekstualni sadržaj poruke
 - ▶ `$mime->setTXTBody('poruka');`
- ▶ Metodi za sastavljanje tijela poruke (`get`) je potrebno reći da se koristi `utf-8` za slanje tekstualnih poruka
 - ▶ `$body = $mime->get(array('text_charset' => 'utf-8'));`

Primjer (1)



```
▶ require_once 'Mail.php';
   require_once 'Mail/mime.php';

$text = "Tekstualna poruka\nđšććž\nžĆČĐŠ";
$headers = array('Subject' => 'Tekstualna žććđš');

$mime = new Mail_mime("\n");

$mime->setTXTBody($text);

// obavezan redosljed!!!
$body = $mime->get(array('text_charset' => 'utf-8'));
$headers = $mime->headers($headers);

$mail = Mail::factory('mail');
$mail->send('ime.prezime@fer.hr', $headers, $body);
```

Slanje HTML poruka

- ▶ Kad se šalju HTML poruke, potrebno je slati i sadržaj s čistim tekstom
 - ▶ Zbog klijenata koji ne podržavaju HTML ili im treba iz nekog drugog razloga
 - ▶ Najjednostavniji način je koristiti funkciju `strip_tags`
- ▶ Metodom `setHTMLBody` postavlja se HTML sadržaj poruke
- ▶ Metodi `get` potrebno je reći da se koristi `utf-8` za slanje HTML poruka
 - ▶

```
$body = $mime->get(  
    array('text_charset' => 'utf-8',  
        'html_charset' => 'utf-8'));
```

Primjer (2)



```
▶ $text = 'Tekstualna verzija';
$html = '<html><body>
        <b>HTML</b> verzija<br>
        </body></html>';

$mime->setTXTBody($text);
$mime->setHTMLBody($html);

$body = $mime->get(
    array('text_charset' => 'utf-8',
          'html_charset' => 'utf-8'));
$headers = $mime->headers($hdrs);
```

Prilaganje datoteka

- ▶ Metodom `addAttachment` moguće je priložiti neku datoteku
 - ▶ Prvi parametar je putanja do datoteke
 - ▶ Drugi parametar je mime tip (neobavezan)
- ▶ Primjer:
 - ▶ `$mime->addAttachment('slika.jpg',
 'image/jpeg');`

Ugrađene slike

- ▶ Ugrađene slike nisu priložene na način kao i sve ostale datoteke
 - ▶ Ne pojavljuju se u popisu priloženih datoteka
- ▶ Metodom `addHTMLImage` prilaže se slika koju je moguće ugraditi u HTML
 - ▶ Prvi parametar je putanja do slike
 - ▶ Drugi parametar je mime tip
 - ▶ Treći parametar je ime slike na koje se referencira unutar HTML dijela poruke
- ▶ Primjer:
 - ▶ `$mime->addHTMLImage($file, 'image/jpeg', 'slikica.jpg');`

Primjer (3)



```
▶ $text = 'Tekstualna verzija bez slike';  
$html = '<html><body>  
        
      </body></html>';  
  
$mime->setTXTBody($text);  
$mime->setHTMLBody($html);  
  
$mime->addHTMLImage($file,  
                   'image/jpeg',  
                   'slikica.jpg');
```

Zadatak (1.1)



- ▶ Napišite skriptu pomoću koje posjetitelj može zatražiti informaciju o dostupnosti proizvoda
- ▶ Stranica mora imati obrazac s ovim poljima:
 - ▶ Adresa e-pošte
 - ▶ Ime i prezime
 - ▶ Ime proizvoda
 - ▶ Količina proizvoda
 - ▶ Upit proizvođaču (textarea)
- ▶ Kad posjetitelj pošalje obrazac, potrebno je poslati e-poruku u Vaš sandučić sa svim unesenim informacijama
 - ▶ Samo čisti tekst (ne HTML)
- ▶ Kad riješite zadatak, **onemogućite ga!!!**

Zadatak (1.2)



- ▶ Prepravite prethodni zadatak tako da šalje i HTML poruke
- ▶ Koristite Smarty za generiranje tekstualnih i HTML poruka!
 - ▶ Ista instanca klase Smarty, ali različiti predlošci
 - ▶ Manje posla
 - ▶ Metodom `fetch` (umjesto `display`) moguće je dohvatiti rezultat spajanja predloška s podacima
- ▶ Kad riješite zadatak, **onemogućite ga!!!**

Zadatak (1.3)



- ▶ Stranicu s obrascem prepravite tako da koristi Smarty i SmartyValidate
 - ▶ Adresa e-pošte mora biti ispravna
 - ▶ Ime i prezime ne smije biti prazno
 - ▶ Ime proizvoda ne smije biti prazno
 - ▶ Količina proizvoda mora biti unutar intervala [1..10]
 - ▶ Upit proizvođaču ne smije biti dulji od 255 znakova
- ▶ Kad riješite zadatak, **onemogućite ga!!!**