

Osnove izrade PHP aplikacija Posluživanje, RSS

Izradio: Davor Cihlar

Plan

- ▶ Što ćemo naučiti
 - ▶ Kako izraditi vlastite internetske usluge
 - ▶ RSS

Kako izraditi vlastite internetske usluge

- ▶ Nije puno drugačije od običnih stranica
- ▶ Koristi se HTTP protokol, ali XML ili JSON kao sadržaj
 - ▶ Zaglavlje Content-Type **nije** text/html!
- ▶ Parametre je najjednostavnije primiti preko URL-a
 - ▶ Moguće i preko POST, PUT, DELETE (složenije)
- ▶ U jednoj je skripti moguće podržati i više funkcija (API-ja)
 - ▶ API (engl. application programming interface) je sučelje između različitih programa

Nadograđivanje internetskih usluga

- ▶ Ako su internetske usluge javne, nije poželjno njihovo nadograđivanje
 - ▶ Nadogradnjom svim ostalim korisnicima stare aplikacije više neće raditi!
- ▶ Poželjno je:
 - ▶ Dodati novu funkciju i ostaviti staru
 - ▶ Dodati poseban parametar prilikom poziva funkcije koji određuje o kojoj inačici funkcije se radi
 - ▶ Ukoliko parametar nije zadan, radi se o prvoj inačici
- ▶ Nikako nije poželjno mijenjati postojeće funkcije na takav način da bi se mogla narušiti kompatibilnost

Primjer (1)



▶ Usluga trenutnog vremena (JSON)

```
▶ $timestamp = isset($_GET['timestamp']) ?  
    (int)$_GET['timestamp'] : time();
```

```
$data = getdate($timestamp);
```

```
$json = json_encode($data);
```

```
header('Content-Type: application/json');
```

```
header('Content-Length: ' . strlen($json));
```

```
echo $json;
```

Primjer (2.1)



- ▶ Podrška za više različitih API-ja
 - ▶ Poziv na udaljenu proceduru RPC (*remote procedure call*)

```
▶ function vrijeme() {  
    return getdate();  
}
```

```
function greska() {  
    throw new Exception('Kao da je bila greška', 42);  
}
```

```
$api = $_GET['action'];  
$result = $api();  
$json = json_encode($result);
```

```
header('Content-Type: application/json');  
header('Content-Length: ' . strlen($json));  
echo $json;
```

Primjer (2.2)



- ▶ Što su problemi u prijašnjoj skripti?
 - ▶ Nepostojanje dojave greške
 - ▶ Sigurnost
- ▶ Sigurnost možemo dobiti ovako:
 - ▶

```
$dopusteni = array('vrijeme', 'greska');
```

```
if (!in_array($api, $dopusteni, true))  
    throw new Exception('Zabranjen API!', 403);
```
- ▶ Greške možemo javljati preko zaglavlja
 - ▶ Npr. "403 Forbidden" ili "400 Bad request"
- ▶ Ili kao dio rezultata
 - ▶ Npr. rezultat je asocijativno polje i element "error" je rezerviran za kod greške

Primjer (2.3)



- ▶ Greške elegantno možemo dojavljivati pomoću try...catch bloka

```
▶ try {  
    if (!isset($_GET['action']))  
        throw new Exception('API nije definiran!', 400);  
  
    $api = $_GET['action'];  
    ... # pozivanje API-ja  
    $result = array_merge($result,  
                          array('error'=>0));  
} catch (Exception $e) {  
    $result = array('error' => $e->getCode(),  
                  'errorMessage' => $e->getMessage());  
}  
  
$json = json_encode($result);  
... # ispis JSON-a
```

Primjer (2.4)



- ▶ PHP-ove greške i upozorenja mogu biti problem s izlaznim formatom (JSON/XML)
 - ▶ Npr. dođe do greške u skripti, PHP ispiše grešku i nakon toga JSON ili XML više nije moguće učitati
 - ▶ Kako vidjeti do koje greške je došlo?!
- ▶ Rješenje:

```
function izbaciGresku($errno, $errstr) {  
    throw new Exception($errstr, $errno);  
}  
  
set_error_handler('izbaciGresku', E_ALL);
```

Prijava

- ▶ Neke usluge ovise o prijavljenom korisniku
 - ▶ Npr. pristup privatnom kalendaru
- ▶ Moguće koristiti sjednice, ali nekad su problem kolačići
 - ▶ Potrebno omogućiti podršku za kolačiće na klijentu
- ▶ Često se koristi ključ kao parametar usluge
 - ▶ Do ključa je prvo potrebno doći preko usluge za prijavu
 - ▶ Taj ključ može biti isti onaj od sjednice

Primjer (3)



- ▶ Umjesto samo `session_start` sad se koristi:

```
▶ if (isset($_GET['key'])) {  
    session_id($_GET['key']);  
    session_start();  
}
```

- ▶ Skripta stvara (tj. u ovom slučaju samo nastavlja) sjednicu ukoliko je zadan parametar "key"

- ▶ Ne koriste se kolačići!

- ▶ Funkcija za prijavu bi trebala biti nešto poput:

```
▶ function prijava() {  
    if ($_GET['user'] !== 'pero' ||  
        $_GET['pass'] !== '123')  
        throw new Exception('Invalid login', 403);  
    session_start();  
    $_SESSION['user'] = $_GET['user'];  
    return array('key' => session_id());  
}
```

Zadatak (1)



- ▶ Napravite internetsku uslugu koja ima funkciju "whoAmI" (engl. tko sam ja)
 - ▶ Funkcija vraća informacije o prijavljenom korisniku (dovoljno samo korisničko ime)
 - ▶ Koristiti prijavu kako je prije objašnjeno
- ▶ Za demonstraciju nije potrebno raditi klijenta
 - ▶ Prvo se posjeti link: `?action=login&user=X&pass=Y`
 - ▶ Zatim: `?action=whoAmI&key=KLJUČ`
 - ▶ X = korisničko ime
 - ▶ Y = zaporka
 - ▶ KLJUČ = ključ dobiven prilikom prijave

RSS

- ▶ Dolazi od engl. Really Simple Syndication
- ▶ Omogućuje korisniku centralizirano prikupljanje obavijesti (npr. novosti) s više poslužitelja
- ▶ Moguće je filtrirati obavijesti na temelju parametara iz URL-a
 - ▶ Poslužitelj dohvaća samo informacije potrebne korisniku
- ▶ Temeljen na XML-u
- ▶ Preporučeni preglednici:
 - ▶ Firefox, RSSOwl (multiplatform)
 - ▶ Google reader (online)
 - ▶ Liferea (Gnome), Akregator (KDE)
- ▶ http://www.w3schools.com/rss/rss_intro.asp

RSS – primjer

- ▶ <http://www.fer.hr/feed/rss.php?url=/>
- ▶

```
<?xml version="1.0" encoding="UTF-8" ?>
<rss version="2.0">
<channel>
  <title>FER</title>
  <link>http://www.fer.hr/</link>
  <description>FER Feed</description>
  <language>hr</language>
  <ttl>240</ttl>

  <item>
    <title>Laboratorij za podvodne ...</title>
    <pubDate>Tue, 05 Apr 2011 11:47:06 GMT</pubDate>
    <description>&lt;table border="0"&gt;
      ...</description>
    <link>http://www.fer.hr/?@=25c6d#news_8980</link>
    <guid>http://www.fer.hr/?@=25c6d#news_8980</guid>
  </item>
  ...
```

RSS – tagovi

- ▶ **Bitni tagovi u zaglavlju kanala:**
 - ▶ `title` – naslov kanala (obavezan)
 - ▶ `link` – početna adresa stranice (obavezan)
 - ▶ `description` – opis kanala (obavezan)
 - ▶ `ttl` – preporučeno vrijeme osvežavanja u minutama
 - ▶ `item` – po jedan za svaku obavijest
- ▶ **Bitni tagovi u obavijesti (unutar `item`):**
 - ▶ `title` – naslov obavijesti (obavezan)
 - ▶ `link` – adresa obavijesti (obavezan)
 - ▶ `description` – sadržaj obavijesti (obavezan)
 - ▶ `guid` – jedinstvena vrijednost za svaku obavijest
 - ▶ `pubDate` – posljednje vrijeme objave
 - ▶ `author` – e-mail autora obavijesti

guid

- ▶ RSS vraća sve ili prvih nekoliko obavijesti
 - ▶ Kako znati koje su nove?
- ▶ `guid` omogućuje klijentu da zapamti koje obavijesti su pročitane
- ▶ Bez `guid`, nakon što se RSS osvježi, ne bi bilo moguće pratiti koja vijest je pročitana, a koja nije
- ▶ `guid` može biti npr. ID iz baze ili sažetak (`md5` ili `sha1`) sadržaja obavijesti

pubDate

- ▶ `pubDate` je datum objave obavijesti
- ▶ Mora biti zapisan u RFC822 formatu!
 - ▶ Primjer:
 - ▶ `Sun, 17 Apr 2011 13:19:21 +0200`
- ▶ Pomoću funkcije `date()` možemo pretvoriti *timestamp* u potrebni format
 - ▶ `date('r', $timestamp)`
 - ▶ *Timestamp* je broj sekundi prošlih od 1.1.1970
- ▶ Pomoću funkcije `time()` možemo dobiti trenutni *timestamp*

Generiranje RSS-a

- ▶ RSS, tj. XML je moguće generirati na više načina
 - ▶ Preporučamo direktno generiranje ispisom (echo)

- ▶ Primjer:

```
<?php
header('Content-Type: text/xml');
echo '<?xml version="1.0" encoding="UTF-8" ?>';
?>
```

```
<rss version="2.0">
<channel>
  <title>Moj RSS</title>
  <link>http://php.fer.hr/...</link>
  <description>Moj prvi RSS</description>
<?php ... generiranje ostatka ... ?>
</channel>
</rss>
```

RSS s HTML obavijestima

- ▶ RSS podržava HTML obavijesti
- ▶ U opis od pojedine obavijesti (`description`) ne može doći izvorni HTML
 - ▶ Potrebno je sve "`<`" i "`>`" zamijeniti s "`<`" i "`>`"
 - ▶ Moguće je koristiti i funkciju `htmlspecialchars`
- ▶ **Podsjetka:**
 - ▶ XML je jako osjetljiv na greške! Neki RSS preglednici neće niti javiti gdje je došlo do greške.
 - ▶ Greške u XML-u možete pronaći tako da napravite novu skriptu i u njoj jednostavno učitate problematičnu XML stranicu pomoću funkcije `simplexml_load_file`
 - ▶ Funkcija će ispisati sve nepravilnosti koje pronađe

Referenciranje na RSS

- ▶ Stranice se mogu referencirati na RSS
 - ▶ Preglednik (npr. Firefox) detektira tu referencu i prikaže RSS ikonu na koju je moguće kliknuti
- ▶ HTML kod:
 - ▶ `<link rel="alternate" type="application/rss+xml" title="NASLOV" href="ADRESA">`
 - ▶ NASLOV - naslov kanala
 - ▶ ADRESA - adresa na kojoj se nalazi RSS



Zadatak (2)



- ▶ Napišite RSS skriptu koja javlja obavijesti o dodanim HTML datotekama
 - ▶ Datoteke se nalaze u mapi "obavijesti"
 - ▶ Nije potrebno raditi skriptu za postavljanje datoteka
 - ▶ Moguće ih je izlistati ovako:
 - ▶ `glob('obavijesti/*.html')`
- ▶ Svaka datoteka je zasebna obavijest
 - ▶ Naslov obavijesti je ime datoteke
 - ▶ Link obavijesti je link na datoteku
 - ▶ Sadržaj obavijesti je sadržaj datoteke
 - ▶ Vrijeme obavijesti je vrijeme modifikacije datoteke
 - ▶ `date('r', filemtime($datoteka))`
 - ▶ `guid` je sažetak (md5 ili sha1) sadržaja datoteke

Zadatak (3)



- ▶ Napravite stranicu s vijestima (običan `var_dump`) koja se referencira na RSS
 - ▶ Ubacite i običan link na RSS
- ▶ RSS treba sadržavati samo prvih 10 vijesti
- ▶ Vijesti se dohvaćaju iz baze, a dodaju ručno
- ▶ Naslov vijesti je prvih 10 znakova sadržaja
 - ▶ Koristiti `strip_tags` kako HTML tagovi ne bi došli u naslov
- ▶ Link vijesti vodi na početnu stranicu, ali s dodatnim GET parametrom `vijest=ID`
 - ▶ Nije potrebno implementirati takav prikaz na stranici
- ▶ Koristiti sažetak ili ID kao `guid`
- ▶ Ne zaboraviti na `htmlspecialchars`