Principi softverskog inženjerstva

Vežbe - IX nedelja - PHP i Ajax

Dražen Drašković, asistent
Nenad Vitorović, asistent
Elektrotehnički fakultet
Univerziteta u Beogradu

Aplikacija

- Aplikacija je bilo koji program koji je napravljen da bi bio pojednostavljen ili obavljen neki zadatak
- Klasična aplikacija je instalirana na klijentskoj mašini
- Klasična aplikacija se pokreće izvršavanjem određenog izvršnog fajla (exe)

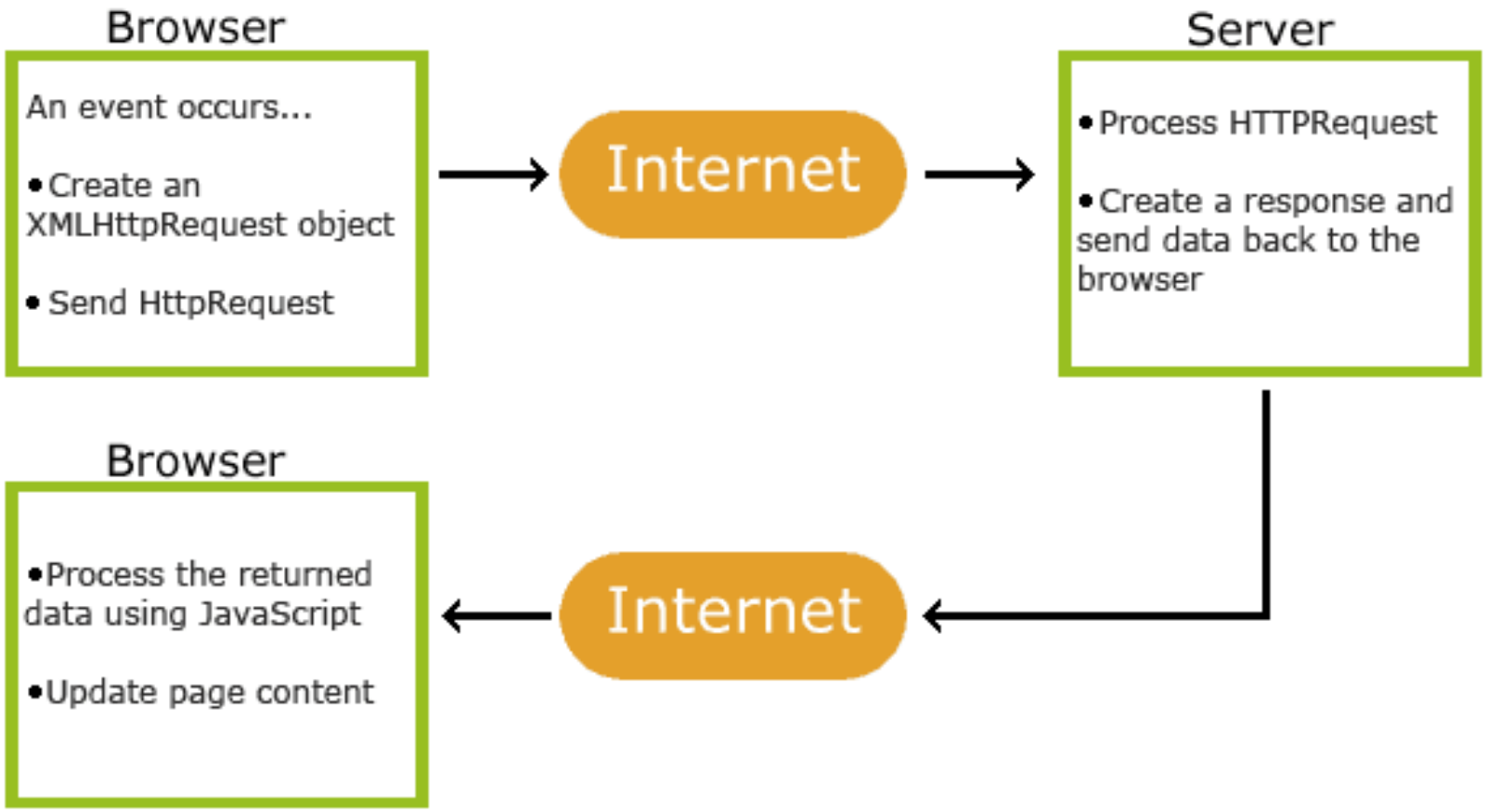
Web 2.0

- Web 2.0 je termin skovan u O'Reilly Media
- Podrazumeva da korisnik nije samo konzument sadržaja, već je i kreator sadržaja (*producer*)
- Podrazumeva se da korisnici aktivno kreiraju
  - Slobodna klasifikacija sadržaja (kao na Wiki)
  - Bogat i fleksibilan UI interfejs – kompletna aplikacija u browser-u (AJAX)
  - Korisnik kao kontributor
  - Dostupnost sadržaja svima
  - Poverenje: kontribucije dostupne i vidljive svima ili kontrolisanoj masi ljudi
  - Masovno učešće

AJAX

- AJAX = **Asynchronous JavaScript and XML**
- AJAX je tehnika za kreiranje brzih i dinamičkih veb stranica. AJAX nije tehnologija, već skup tehnologija!
- AJAX dozvoljava veb stranicama da se menjaju asinhrono izmenom male količine podataka
- Komunikacija sa serverom odvija u pozadini i na taj način je moguće menjati delove stranice, a ne celu stranicu
- Klasične veb stranice (koje ne koriste AJAX) moraju menjati celu stranicu ako se bilo koji deo stranice promeni (*reload*)
- Primeri aplikacija koje koriste AJAX: Google Maps, Gmail, YouTube, Facebook…

AJAX


Browser
An event occurs...
• Create an XMLHttpRequest object
• Send HttpRequest
Internet
Server
• Process HTTPRequest
• Create a response and send data back to the browser
Internet
Browser
• Process the returned data using JavaScript
• Update page content

Standardi

- AJAX je baziran na Internet standardima i koristi kombinaciju:
  - **XMLHttpRequest** objekta
    (da asinhrono razmenjuje podatke sa serverom)
  - **JavaScript/DOM**
    (da prikaže/poveže se sa informacijama)
  - **CSS** (za prikaz stila podataka)
  - **XML** (najčešće korišćen za format prenetih podataka)
- AJAX aplikacije su nezavisne po pitanju čitača i platformi!
  - Nebitno je šta se nalazi na serveru
  - Nebitno je koji je browser u pitanju – trebalo bi da funkcioniše na svakom

Google Suggest

- AJAX je postao popularan od 2005.godine kada je počeo da ga koristi Google, za opciju Google Suggest.
- Google Suggest koristi AJAX da kreira veoma dinamički veb interfejs:
  Kada se počne sa unosom u Google search box, JavaScript šalje karaktere serveru i server vraća listu sugestija.
- Osnovna komponenta AJAX tehnologije je XMLHttpRequest objekat.

XMLHttpRequest

- Svi moderni čitači podržavaju XMLHttpRequest objekat (IE5 i IE6 koriste ActiveXObject).
- XMLHttpRequest objekat se koristi za razmenu podataka sa serverom u pozadini. Na ovaj način je moguće promeniti deo stranice, bez učitavanja cele stranice.
- Ostali čitači (IE7+, Firefox, Chrome, Safari i Opera) imaju ugrađen XMLHttpRequest objekat.
- Syntaksa za kreiranje XMLHttpRequest objekta je:

```
xmlhttp=new XMLHttpRequest();
```

- Starije verzije Internet Explorer (IE5 i IE6) koriste ActiveX:

```
xmlhttp=new ActiveXObject("Microsoft.XMLHTTP");
```

XMLHttpRequest

- Da bi se obuhvatili svi moderni čitači, uključujući IE5 i IE6, treba proveriti da li čitač podržava XMLHttpRequest objekat.
- Ako podržava, kreira se XMLHttpRequest objekat, ako ne, kreira se ActiveXObject:

```
if (window.XMLHttpRequest){
  // code for IE7+, Firefox, Chrome,
  Opera, Safari
    xmlhttp=new XMLHttpRequest();
}
else{// code for IE6, IE5
xmlhttp=new
  ActiveXObject("Microsoft.XMLHTTP");
}
```

XMLHttpRequest

- Da bi se poslao zahtev ka serveru, koriste se `open()` i `send()` metode XMLHttpRequest objekta:

```
xmlhttp.open("GET","ajax_info.txt",true);
xmlhttp.send();
```

- `open(`*`method,url,async`*`)`
- Specificira tip zahteva, URL, i da li se zahtev izvršava asinhrono ili ne.
- *method*: tip zahteva: GET ili POST
  url: lokacija fajla na serveru
  async: true (asynchronous) ili false (synchronous)
- `send(`*`string`*`)`
- Šalje se zahtev ka serveru.
  string: Koristi se samo za POST zahteve

GET ili POST

- GET je jednostavniji i brži nego POST, i koristi se u većini slučajeva.
- Ipak uvek treba koristiti POST kada:
  - Keširani fajl nije potreban (promeniti fajl ili bazu podataka na serveru)
  - Šalje se velika količina podataka na server (POST nema limit u veličini)
  - Šalje se unos korisnika (koji može sadržati nepoznate karaktere), POST je mnogo više robustan i bezbedniji nego GET

GET

- Jednostavan GET zahtev:

```
xmlhttp.open("GET","demo_get.php",true);
xmlhttp.send();
```

- Može se slati jedistveni ID sa URL-om da bi se izbeglo keširanje koje browser vrši:

```
xmlhttp.open("GET","demo_get.php?t=" +
  Math.random(),true);
xmlhttp.send();
```

- Za slanje dodatnih informacija, one se dodaju na kraj URL-a:

```
xmlhttp.open("GET","demo_get2.php?fname=
  Henry&lname=Ford",true);
xmlhttp.send();
```

POST

- JednostavanPOST zahtev:

```
xmlhttp.open("POST","demo_post.php",true);
xmlhttp.send();
```

- Za POST podatke je potrebno dodati HTTP header sa `setRequestHeader()`, pa se specificiraju podaci koji se žele poslati

```
xmlhttp.open("POST","ajax_test.php",true);
xmlhttp.setRequestHeader("Content-type",
  "application/x-www-form-urlencoded");
xmlhttp.send("fname=Henry&lname=Ford");
```

- setRequestHeader(*header,value*)
- Dodaje HTTP headers zahtevu.
  header: specificira ime headera
  value: specificira vrednost headera

URL

- URL predstavlja fajl na serveru
- url parametar `open()` metoda je adresa fajla na serveru:
  - `xmlhttp.open("GET","ajax_test.php",true);`
- Fajl može biti bilo koje vrste,
  na primer .txt ili .xml (statički fajl)
  ili server skripting fajlovi kao što su .asp ili .php
  (koji mogu izvršavati akcije na serveru pre nego što se generiše odgovor).

Asinhronost

- AJAX predstavlja **Asynchronous JavaScript and XML**, i da bi se XMLHttpRequest objekat ponašao kao AJAX, async parametar open() metoda treba da bude postavljen na true:
  - `xmlhttp.open("GET","ajax_test.php",true);`
- Slanje asinhronog zahteva je veliki dobitak za Web programere.
- Mnogi zadaci koji se izvršavaju na serveru su vremenski zahtevni.
- Pre AJAX-a,ove operacije su prouzrokovale da se aplikacija blokira.
- Sa AJAX-om, JavaScript ne treba da čeka na odgovor servera, već može da:
  - Izvršava druge skriptove, dok čeka na odgovor servera
  - Obrađuje odgovor kada stigne

Async = true

- Kada se koristi async=true, specificira se call-back funkcija koja se izvršava kada je odgovor spreman u okviru onreadystatechange događaja:

```
xmlhttp.onreadystatechange = function(){
   if (xmlhttp.readyState==4 &&
   xmlhttp.status==200){ // OK status
       document.getElementById("myDiv").
        innerHTML=xmlhttp.responseText;
   }
}
xmlhttp.open("GET","ajax_info.txt",true);
xmlhttp.send();
```

Async = false

- Da bi se koristio async=false, treba promeniti treći parametar u open() metodu na false:

```
xmlhttp.open("GET","ajax_info.txt",false);
```

- Upotreba async=false se ne preporučuje, ali za nekoliko manjih zahteva je u redu.
- Treba zapamtiti da se JavaScript NEĆE nastaviti da se izvršava , sve dok ne dođe odgovor od servera.
- **Napomena:**
  Kada se koristi async = false, ne treba pisati onreadystatechange funkciju - samo treba nastaviti kod posle send() naredbe:
  -
```
xmlhttp.open("GET","ajax_info.txt",false);
xmlhttp.send();
document.getElementById("myDiv").innerHTML
=xmlhttp.responseText;
```

Odgovor servera

- Da bi se dobio odgovor od servera koristi se responseText ili responseXML property od XMLHttpRequest objekta.
  - responseText - dobija se odgovor u formi Stringa
  - responseXML - dobija se odgovor kao XML objekat
- responseText Property – ako odgovor nije XML koristi se ovaj property.
  - `document.getElementById("myDiv").innerHTML = xmlhttp.responseText;`
- responseXML Property ako odgovor jeste XML i ako se želi da se parsira kao XML objekat
  - ```
    xmlDoc=xmlhttp.responseXML;
    var txt="";
    x=xmlDoc.getElementsByTagName("ARTIST");
    for (i=0;i<x.length;i++){
      txt=txt + x[i].childNodes[0].nodeValue +
    "<br />";
    }
    document.getElementById("myDiv").innerHTML =
    txt;
    ```
    ```

onreadystatechange

- Kada se šalje zahtev serveru, potrebno je izvršiti neke akcije na osnovu odgovora
- onreadystatechange događaj se ostvari svaki put kada se readyState promeni.
- readyState property sadrži status XMLHttpRequest.
- Tri bitna atributa XMLHttpRequest objekta su:
- onreadystatechange – pamti funkciju (ili ime funkcije) koja će se zvati automatski kada se readyState property promeni
  - readyState – sadrži status XMLHttpRequest :
    0: Zahtev nije inicijalizovan
    1: uspostavljena je veza sa serverom
    2: zahtev primljen
    3: zahtev se obrađuje
    4: zahtev završen i odgovor spreman
- status
  - 200: "OK"
    404: Page Not Found
    500: Internal Server Error
- U okviru onreadystatechange događaja, specificira se šta će se desiti kada je zahtev servera spreman za obradu
- Kada je readyState 4 i status je 200, odgovor je spreman i nema greške
- I ostale statuse bi trebalo obraditi, kako bi korisnik imao povratnu informaciju o greški

onreadystatechange

- ```
  xmlhttp.onreadystatechange=function()
    {
    if (xmlhttp.readyState==4 &&
        xmlhttp.status==200)
      {
      document.getElementById("myDiv").
          innerHTML=xmlhttp.responseText;
      }
    }
  ```
- onreadystatechange događaj će se pozvati četiri puta po jednom za svaku promenu readyState.
  ```

callback funkcija

- callback funkcija se šalje kao parametar drugoj funkciji.
- Ako postoji više AJAX obrada na jednom sajtu, potrebno je kreirati jednu standardnufunkciju za kreiranje XMLHttpRequest objekta, i pozivati je za svaki AJAX obradu.
- Poziv funkcije treba da sadrži URL i šta raditi na onreadystatechange (što je različito ua svaki poziv):
  -
```
function myFunction()
{
loadXMLDoc("ajax_info.txt",function()
  {
  if (xmlhttp.readyState==4 &&
      xmlhttp.status==200){

      document.getElementById("myDiv").
      innerHTML=xmlhttp.responseText;
    }
  });
}
```

Primer

- Kada korisnik upiše karakter u tekst polje, pozove se "showHint()”.
  Ova funkcija je vezana na "onkeyup”:

```
function showHint(str)
    {
    if (str.length==0)
      {  // no suggestion
        document.getElementById("txtHint").innerHTML="";
        return;
      }
    if (window.XMLHttpRequest)
      {// code for IE7+, Firefox, Chrome, Opera, Safari
        xmlhttp=new XMLHttpRequest();
      }
    else
      {// code for IE6, IE5
        xmlhttp=new ActiveXObject("Microsoft.XMLHTTP");
      }
    xmlhttp.onreadystatechange=function()
      {
      if (xmlhttp.readyState==4 && xmlhttp.status==200)
        {
        document.getElementById("txtHint").innerHTML=xmlhttp.responseText;
        }
      }
    xmlhttp.open("GET","gethint.php?q="+str,true);
    xmlhttp.send();
    }
```

Primer

- Ako je tekst polje prazno (str.length==0), funkcija briše sadržaj polja txtHint i napušta se funkcija.
- Ako tekst polje nije prazno, showHint() funkcija izvršava sledeće:
  - Kreira se XMLHttpRequest objekat
  - Kreira se funkcija koje će se izvršiti kada odgovor servera bude spreman
  - Šalje se zahtev fajlu na serveru
  - Obratiti pažnju da je parametar (q) dodat na URL (sa sadržajem tekst polja)

Primer php fajl

```
<?php
// Popunjavamo niz sa imenima
$a[]="Ana";
$a[]="Bojana";
$a[]="Marina";
$a[]="Dijana";
$a[]="Ena";
$a[]="Marija";
$a[]="Gordana";
$a[]="Milena";
$a[]="Ivana";
$a[]="Jasna";
$a[]="Biljana";
$a[]="Ljiljana";
$a[]="Nevena";
$a[]="Olivera";
$a[]="Petra";
$a[]="Aleksandra";
$a[]="Danijela";
$a[]="Jovana";
$a[]="Doris";
$a[]="Katarina";
$a[]="Emilija";
$a[]="Sanja";
$a[]="Tamara";
$a[]="Una";
$a[]="Violeta";
$a[]="Natasa";
$a[]="Tijana";
$a[]="Elena";
$a[]="Vukica";
$a[]="Vesna";

//dobijamo parametar q iz URL adrese
$q=$_GET["q"];
```

Primer php fajl

```
//ako je duzina q>0, dobijamo hintove za taj parametar
if (strlen($q) > 0)
  {
  $hint="";
  for($i=0; $i<count($a); $i++)
    {
    if (strtolower($q)==strtolower(substr($a[$i],0,strlen($q))))
      {
      if ($hint=="")
        {
        $hint=$a[$i];
        }
      else
        {
        $hint=$hint." , ".$a[$i];
        }
      }
    }
  }

// Postavljamo izlaz na "nema sugestija"  ako nije pronadjen hint ili korektna vrednost
if ($hint == "")
  {
  $response="nema sugestija";
  }
else
  {
  $response=$hint;
  }

//izlaz
echo $response;
?>
```

Primer 2

id	FirstName	LastName	Age	Hometown	Job
1	Jelica	Protic	48	Beograd	vanredni profesor
2	Bosko	Nikolic	38	Valjevo	vanredni profesor
3	Nemanja	Kojic	25	Loznica	asistent
4	Drazen	Draskovic	24	Beograd	asistent

Primer 2

```
<html>
<head>
<script type="text/javascript">
function showUser(str) {
if (str=="") {
     document.getElementById("txtHint").innerHTML="";
     return;
  }
if (window.XMLHttpRequest) { // code for IE7+, Firefox, Chrome, Opera,
Safari
     xmlhttp=new XMLHttpRequest();
  }
else { // code for IE6, IE5
     xmlhttp=new ActiveXObject("Microsoft.XMLHTTP");
  }
xmlhttp.onreadystatechange=function() {
  if (xmlhttp.readyState==4 && xmlhttp.status==200) {
     document.getElementById("txtHint").innerHTML=xmlhttp.responseText;
   }
 }
xmlhttp.open("GET","getuser.php?q="+str,true);
xmlhttp.send();
}
</script>
</head>
```

Može biti HTML

Primer 2

```
<body>

<form>
<select name="users"
onchange="showUser(this.value)">
    <option value="">Odaberite osobu:</option>
    <option value="1">Jelica Protic</option>
    <option value="2">Bosko Nikolic</option>
    <option value="3">Nemanja Kojic</option>
    <option value="4">Drazen Draskovic</option>
</select>
</form>
<br />
<div id="txtHint"><b>Informacije o osobi ce
biti prikazane ovde.</b></div>

</body>
</html>
```

Primer 2

- showUser() funkcija izvršava sledeće:
  - Proverava da li je osoba izabrana
  - Kreira XMLHttpRequest objekat
  - Kreira funkciju koja će se izvršiti kada odgovor servera bude spreman
  - Šalje zahtev fajlu na serveru
  - Obratiti pažnju da je parametar (q) dodat na URL (sa sadržajem dropdown liste)

Primer 2

```
<?php
$q=$_GET["q"];

$con = mysql_connect('localhost', 'peter', 'abc123');
if (!$con){
  die('Could not connect: ' . mysql_error());
}

mysql_select_db("ajax_demo", $con);

$sql="SELECT * FROM user WHERE id = '".$q."'";

$result = mysql_query($sql);

echo "<table border='1'>
<tr>
      <th>Firstname</th>
      <th>Lastname</th>
      <th>Age</th>
      <th>Hometown</th>
      <th>Job</th>
</tr>";
```

Primer 2

```
while($row = mysql_fetch_array($result)){
   echo "<tr>";
   echo "<td>".$row['FirstName']."</td>";
   echo "<td>".$row['LastName'] . "</td>";
   echo "<td>".$row['Age']."</td>";
   echo "<td>".$row['Hometown']."</td>";
   echo "<td>".$row['Job']."</td>";
   echo "</tr>";
   }
echo "</table>";

mysql_close($con);
?>
```

Primer 2

- Kada je upit poslat od strane JavaScript ka PHP fajlu, izvrši se:
  - PHP otvara konekciju ka MySQL serveru
  - Korektna osoba se pronađe
  - HTML tabela se kreira, popunjena podacima i šalje se dalje "txtHint" polju

AJAX u JS bibliotekama

- Svaki ozbiljniji i privlačniji sajt koristi AJAX
- Dosta ponavljajućih delova (*eng. boiler-plate*)
- Biblioteke: JQuery, MooTools, Dojo, YUI…
  - Pripremljeno, spremno za korišćenje
  - Testirano, verifikovano, održavano i prati promene u tehnologiji
  - Mnogo jednostavnije za pisanje
  - Masovno se koristi u industriji

Primer: JQuery

```
$.ajax({
   type: "POST",
   url: „phonebook.php",
   data: {
      name: "John",
      location: "Boston"
   }
}).done(function( result_html) {
      $( "#results").append(result_html);
});
```

Javascript objekat
(JSON: JavaScript Object Notation)