6. laboratorijska vježba: JavaScript i XMLHttpRequest

Cilj vježbe

Izrada Web stranice s elementima promjenjivim na strani klijenta bez osvježavanja stranice, uporaba objekta *XMLHttpRequest* za preuzimanje dodatnih podataka s poslužitelja (AJAX) te korištenje javno dostupnih biblioteka za bržu izradu stranice.

Priprema za vježbu

Proučiti:

- Osnove JavaScripta
  - W3Schools – JavaScript
    - uključivanje vanjske JavaScript datoteke u HTML
    - rad s varijablama, funkcije, događaji

    http://www.w3schools.com/js/
- Osnove koncepta DHTML
  - W3Schools – DHTML
    - koncept, HTML DOM, *event handleri*, promjena stilova elemenata

    http://www.w3schools.com/dhtml/
- Osnove pristupanja i promjene elemenata HTML-a pomoću modela DOM
  - Gecko DOM Reference:Introduction
    - osnovne metode i svojstva

    http://developer.mozilla.org/en/docs/Gecko_DOM_Reference:Introduction#Core_Interfaces_in_the_DOM
- Rad s objektom *XMLHttpRequest*
  - Ajax tutorial

  http://www.xul.fr/en-xml-ajax.html#ajax-in-depth
  - Dynamic HTML and XML: The *XMLHttpRequest* Object

  http://developer.apple.com/internet/webcontent/xmlhttpreq.html

Zadatak za vježbu

Potrebno je nadopuniti postojeću Web stranicu prikazom rezultata pretraživanja dinamičkim elementima na strani preglednika, korištenjem JavaScripta. Proširiti funkcionalnost stranice za prikaz rezultata pretrage prikazom dodatnih detalja o pojedinom osnovnom elementu (ovisno o inačici: osobi, knjizi, CD-u, ...) korištenjem objekta *XMLHttpRequest*.

Dodatno, u dijelu vježbe potrebno je upotrijebiti neku od vanjskih biblioteka JavaScript funkcija za definiranu namjenu, kao pomoć u bržoj izradi aplikacije.

Za rješavanje zadatka bit će potrebna izrada dodatne skripte u jeziku PHP, te određene (manje) promjene u CSS-u, DTD-u i Java servletu iz prethodnih vježbi.

Vježba sadrži dva dijela:

1. Na stranici rezultata pretrage potrebno je **prelaskom strelice miša preko pojedinog retka** tablice **prikazati *tooltip* s dodatnim informacijama** ("balončić", *tooltip*) **o**

osnovnom elementu čiji podaci su prikazani u tom retku. Za prikazivanje *tooltipa* koristiti neku od javno dostupnih biblioteka te namjene.

Primjer *tooltipa* možete vidjeti prelaskom miša preko datuma objave bilo koje obavijesti na Webu FER-a.

2. Korištenjem objekta *XMLHttpRequest* (koji omogućuje komunikaciju s poslužiteljem, slanje zahtjeva i dobivanje odgovora bez osvježavanja cijele stranice - AJAX) **dohvatiti druge dodatne podatke** o odgovarajućem osnovnom elementu i **prikazati ih** na stranici. Za to je potrebno u tablicu rezultata pretraživanja na kraj **dodati još jedan stupac**, naslova **"Akcija"** koji će s u svakom retku sadržavati neki aktivni element (npr. možete odabrati link s tekstom **"Detalji"**, neku prikladnu **sličicu** ili čak **gumb**).

   Klikom mišem na taj element (tekst, sliku ili gumb) **treba poslati zahtjev za dodatnim podacima** prema (novoj) PHP datoteci. PHP datoteka u XML podacima (koje dohvaća od Servleta na isti način kao i stranica prikaza rezultata pretrage) vraća samo dodatne podatke vezane upravo za taj osnovni element. Ova PHP datoteka **vraća odgovor u obliku HTML kôda**, kojeg pomoću JavaScripta i modela DOM treba prikazati na određenom mjestu na stranici rezultata pretrage (novi element <*div*> koji će sadržavati **Dodatne informacije**). Pritom će biti potrebno koristiti JavaScript za promjenu svojstava ili izgleda objekata.

Promjena strukture stranice i podataka

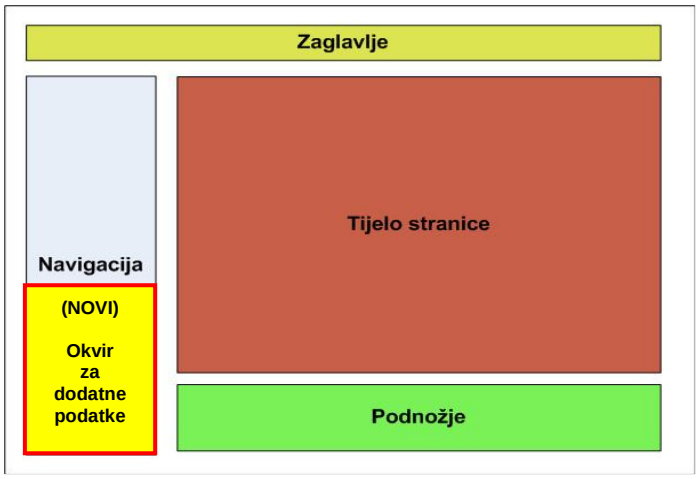
Stranicu rezultata pretrage (koju generira *pretraga.php* iz treće vježbe) treba preurediti na sljedeći način:

- Tablica treba sadržavati **najmanje 3, a najviše 5 stupaca s podacima** koji se prikazuju. Ako ste dosad imali više stupaca potrebno ih je obrisati. Treba **prikazati samo osnovne informacije** (npr. za telefonski imenik: ime, prezime, prvi broj mobitela i sl.) tako da prikaz bude pregledan.

- Tablici treba dodati **novi stupac** naslova **"Akcija"** koji će **u svakom retku** sadržavati neki **aktivni element**, koji će aktivirati **zahtjev za detaljnim informacijama**. Na primjer, možete odabrati link s tekstom **"Detalji"**, "Više o …" neku prikladnu **sličicu** ili čak **gumb**, ovisno o tome koliko vam je određeni element prihvatljiviji. S obzirom da će zahtjev trebati biti prilagođen odgovarajućem osnovnom elementu koji prikazujemo u tom retku i za koji tražimo dodatne podatke, u HTML kôdu i pojedinom pozivu JavaScript funkcije potrebno je dodatno, kao argument, zadati **jedinstveni identifikator elementa** (ovo trenutno ne postoji i time ćemo proširiti stari dio vježbe).

- **Ispod dijela za navigaciju** treba se nalaziti **okvir za dodatne podatke**, koji se ispunjava podacima dobivenim s poslužitelja, kao posljedica **klika na aktivni element** u bilo kojem retku. Okvir za dodatne podatke treba definirati kao **element ** (na sličan način kao područje za navigaciju) i pozicionirati ispod područja u kojem prikazujete navigaciju (kao na slici).

- Svi novi ili promijenjeni elementi trebaju se uklapati u izgled, dizajn i strukturu stranice. Za njihovo uređivanje potrebno je koristiti CSS izrađen u 1. lab. vježbi.



Napomena: Kako bi novom okviru za podatke (<*div*>) mogli jednostavno pristupiti iz JavaScript kôda, pridijelite mu jedinstveni **identifikator** (id).

Za prikaz dodatnih podataka, skripti *detalji.php* (koju ćete dodatno izraditi, objašnjeno kasnije) potrebno je proslijediti **jedinstvenu** oznaku pojedinog elementa koja jedinstveno označava da su dva entiteta ista (npr. kao ISBN za knjige ili JMBG za osobe). Ova oznaka trenutno ne postoji u strukturi XML-a.

U strukturu podataka svakog osnovnog elementa potrebno je **dodati jedinstveni identifikator elementa**. To je **atribut "id"** koji se postavlja kao atribut osnovnog elementa (na drugoj razini od osnovnog elementa). Jedinstveni identifikator može biti (naša preporuka) redni broj, slijedno generiran, tako da svaki element ima broj za jedan veći u odnosu na prethodni element. Pri tome paziti (kao i dosad) da elementi u izvorišnim datotekama podataka o osnovnim elementima mogu imati više redaka, koji u XML zapisu čine jedan osnovni element s jednim jedinstvenim identifikatorom elementa i više podelemenata (npr. knjiga sa više autora).

Potrebno je načiniti sljedeće promjene:

- u **DTD dodati atribut "id"**
- u **Java servletu**, pri generiranju izlaznog XML-a iz tekstualnih datoteka, **dodati osnovnom elementu atribut "id"**. Preporuka: najjednostavnije je da njegova vrijednost bude **broj**, identičan **rednom broju elementa**. Jedan od načina proslijeđivanja broja je proširivanje konstruktora osnovnog razreda atributom "id", kojeg ćete upotrijebiti u metodi ToXML.

Prikaz iskočnog okvira s dodatnim informacijama (balončić, tooltip)

Na stranici rezultata pretrage (skripta *pretraga.php*), potrebno je prilikom **prelaska strelice miša preko pojedinog retka** (odnosno osnovnog elementa), u **iskočnom okviru** (*tooltipu*) **prikazati nekoliko dodatnih informacija o osnovnom elementu** koji je prikazan u tom retku. Te podatke skripta *pretraga.php* dobiva u obliku XML podataka iz odgovora Java Servleta, kojeg se poziva kao i u 5. lab. vježbi. U skripti se treba promijeniti HTML kôd tako da se na prijelaz mišem preko područja retka (*onmouseover*) poziva funkcija za prikaz *tooltipa* uz potrebne argumente.

Napomena: Informacije za prikaz u *tooltipima* se nakon generiranja stranice **moraju nalaziti u kôdu HTML-a**, no neće odmah biti prikazane, već će biti skrivene. Za prikaz ovih informacija nije potrebno ponovno pozivati poslužitelj, **već ih samo prikazati korištenjem JavaScripta.**

U *tooltipu* je **potrebno prikazati najmanje 2, a najviše 4 dodatna podatka** (npr. kod osobe prvi fiksni telefon, kod DVD-a prvi režiser i slično), svaki podatak u novi redak okvira ili odvojen nekim separatorom, tako da piše "tip podatka: vrijednost" (npr. "Telefon: 01 5555555").

Za uspješan prikaz *tooltipa* pri prelasku mišem, potrebno je **registrirati *event handler*, na događaj *onmouseover*,** za **svaki** pojedini element. Obično se na događaj poziva neka JavaScript funkcija, čije se tijelo nalazi u JavaScript (sufiks .js) datoteci. Na primjer:

```
<tr onmouseover="prikaziTooltip('Dodatne informacije', 'green', true)">
   <td>
       Osoba A
   </td>
</tr>
```

Funkcionalnost *tooltipa* nije potrebno samostalno implementirati (ali je dozvoljeno, ako znate i želite ☺). Na Webu postoje mnoge kvalitetne biblioteke JavaScript funkcija, za različite namjene. Korištenje provjerenih, javno dostupnih biblioteka drastično smanjuje vrijeme izrade aplikacije, a često osigurava veću razinu robusnosti i kompatibilnosti (npr. funkcionalnost u različitim preglednicima Weba). Pritom je važno paziti na **autorska prava i licencije biblioteka**!

Predložene biblioteke JavaScript *tooltipova* su:

- DHTML JavaScript Tooltips – Walter Zorn
  http://www.walterzorn.com/tooltip/tooltip_e.htm
- overLIB – Erik Bosrup
  http://www.bosrup.com/web/overlib/
- Yahoo UI Library: Tooltip (**OPREZ!** Za naprednije...)
  http://developer.yahoo.com/yui/container/tooltip

Web stranica svake biblioteke sadrži kratki pripadajući **uvod u korištenje**, kojeg je potrebno proučiti. Obično se radi o **preuzimanju biblioteke, uključivanju .js datoteka** u stranicu, te pozivu JavaScript **funkcije** za prikaz *tooltipa*, uz dodatno slanje argumenata kao što su tekst, boje teksta i pozadine, parametri je li *tooltip* stalno vidljiv ili nestaje nakon pomaka mišem, itd.

Tekst *tooltipa* se funkciji obično može predati i kao HTML kôd. To znači da *tooltip* ne prikazuje samo običan tekst, već HTML sadržaj koji sadrži oznake oblikovanja teksta, tablica i atribute koji koriste CSS stilove nadređene HTML stranice (iz koje pozivate *tooltip*). Moguće ga je predati izravno kao niz znakova (pritom pazite na jednostruke i dvostruke **navodnike**) ili kao JavaScript varijablu (polje, String). *Tooltipove* je potrebno prilagoditi dizajnu vaše Web stranice. Prilagodbu možete obaviti izravno – parametrima pojedinih biblioteka, ili pomoću CSS-a.

Osim predloženih biblioteka, možete koristiti **bilo koju drugu biblioteku sličnih funkcionalnosti**. Pri predaji vježbe nije potrebno znati "kako to interno radi" (ako imate vremena preporučujemo da pogledate kako kôd radi, zbog "opće kulture"). Potrebno je znati objasniti **povezivanje**, **prilagodbu** i **prikazivanje** *tooltipa* i JavaScript biblioteke.

Prikaz detaljnih informacija

Detaljne informacije o pronađenom elementu, koje nisu prikazane u *tooltipu*, dohvaćamo lijevim klikom miša na aktivni element (gumb, sliku ili tekst "Detalji", "Više o …"), u posljednjem stupcu svakog retka tablice rezultata pretrage. Za postizanje bolje uporabivosti Web stranice, koristit ćemo JavaScript objekt *XMLHttpRequest*, temelj principa AJAX (Asynchronous JavaScript and XML) kako bismo izbjegli suvišno osvježavanje stranice.

Namjena objekta *XMLHttpRequest* je **komunikacija s poslužiteljem Weba** (slanje GET ili POST zahtjeva i primanje odgovora) za dohvat **manje količine podataka, bez dohvata cijele stranice i osvježavanja**. Primjer rada s objektom:

```
var req; //deklarirana globalna varijabla za "budući" objekt

function loadXMLDoc(url) {
    if (window.XMLHttpRequest) {                //Firefox, Safari, Opera
        req = new XMLHttpRequest();             //stvaranje novog objekta
    }
    else if (window.ActiveXObject) {    //IE 6+
        req = new ActiveXObject("Microsoft.XMLHTTP");  //ActiveX
    }
    if (req) {
// povezivanje s funkcijom koja će se pozvati pri promjeni stanja zahtjeva
        req.onreadystatechange = napraviNesto;
// GET ili POST, URL skripte, asinkroni (true) ili sinkroni (false) način
        req.open("GET", url, true);
//slanje podataka (null za GET, podaci za POST)
        req.send(null);
        }
    }
}
```

Različiti preglednici Weba imaju različite načine instanciranja ovog objekta. Srećom, nakon instanciranja, funkcije su jednake. Nekoliko objašnjenja:

- **onreadystatechange** – registracija funkcije koja se poziva pri **promjeni stanja zahtjeva**
- **asinkroni / sinkroni način** – u sinkronom načinu, odaziv Web stranice bi **prestao** do dobivanja odgovora, što je obično neželjeni način rada. U asinkronom načinu, zahtjev

se pošalje, izvođenje skripti i rad sa stranicom **se nastavlja**, a specificirana JavaScript funkcija se pozove pri promjeni stanja zahtjeva

- **slanje parametara**
  - za **GET** metodu, parametri se šalju **unutar URL-a** (npr. http://www.fer.hr/detalji.php?id=523&show=simple)
  - za **POST** metodu, parametri se pišu jednako kao i za GET, no šalju se unutar **funkcije *send*** (npr. req.send("id=523&show=simple");)

Napomena: Iz sigurnosnih razloga, objekt *XMLHttpRequest* ne dozvoljava komunikaciju s poslužiteljem na domeni različitoj od domene poslužitelja koji je isporučio Web stranicu (*cross-domain scripting)*. Na primjer, stranica generirana sa http://www.fer.hr ne može poslati upit na http://www.google.com, ali ni na http://rasip.fer.hr. No, moguće je poslati upit na http://www.fer.hr/rasip/ (domene poslužitelja su jednake).

Primjer funkcije koja će se pozvati pri promjeni stanja zahtjeva:

```
function napraviNesto() {
    if (req.readyState == 4) {  //stanje 4 označava primitak odgovora
        if (req.status == 200) {        //kod statusa odgovora - OK
            // ovdje ide neki koristan posao, npr. alert(req.responseText)
        } else {                        //kod statusa nije OK :-(
            alert("Nije primljen 200 OK, nego:\n" + req.statusText);
        }
    }
}
```

Nekoliko objašnjenja:

- funkcija se poziva pri **svakoj** promjeni stanja, zato je potrebno provjeriti je li odgovor doista stigao
- ako stranica nije dostupna ili postoji neki drugi problem, podaci nisu poslani, zato se provjerava status odgovora
- podatke je moguće dohvatiti kao **običan niz znakova**, svojstvom *responseText*, ili kao **XML stablo**, svojstvom *responseXML*

Za rješavanje ovog dijela vježbe potrebno je:

- napisati **skriptu PHP-a *detalji.php*** koja metodom GET prima parametar "id", kao oznaku elementa čije detalje želimo (jedinstveni identifikator elementa, vidjeti promjenu strukture XML-a). Skripta je zapravo smanjenja verzija skripte *pretraga.php*, u kojoj treba dohvatiti kompletnu XML strukturu podataka pozivom Java servleta i putem DOM-a ili XPatha pronaći traženi element prema identifikatoru (3. lab. vježba).
- pripremiti **izlaz** skripte PHP-a u obliku **dijela HTML kôda**, koji sadrži druge dodatne elemenata - informacije koje želimo prikazati (npr. svi brojevi telefona, adrese i e-mailovi; svi autori knjiga, jezik, broj stranica...). Ovaj izlaz iz skripte sadrži **kompletni prikaz s uključenim oznakama HTML-a**, npr. oznake nevidljive tablice, oznake , oznake , klase za poziv stilova CSS-a i slično. Potrebno je prikazati **što je više moguće informacija** (poželjno sve) o nekom elementu koje se nalaze u XML-u (a da se ne poremeti izgled stranice).
- **registrirati događaj *onclick* za svaki pojedini aktivni element**, tako da se pozove vaša funkcija koja će **pripremiti** *XMLHttpRequest* zahtjev (s parametrom!) i **poslati** ga skripti ***detalji.php.***
- u funkciji koja se poziva pri promjeni stanja zahtjeva **preuzeti dobiveni HTML kôd** svojstvom *responseText*

- dobiveni HTML kôd prikazati u okviru **Detaljne informacije**. Nakon svakog zahtjeva za informacijama o novom elementu, potrebno je zamijeniti stare informacije novima.
- **Neobavezni dio vježbe:** za vrijeme čekanja na odgovor, pored aktivnog elementa prikazati tekst **"Tražim"** ili prikazati **ikonu** koja prikazuje status čekanja (npr. *http://en.wikipedia.org/wiki/Image:Spinning_wheel_throbber.gif*). Po primitku odgovora ukloniti tekst / ikonu.

JavaScript i DOM

JavaScript omogućuje upravljanje strukturom modela dokumenta (DOM). Kao i u drugim jezicima s implementacijom DOM-a, potrebno je prvo pronaći i dohvatiti element s kojim želimo raditi, a zatim pročitati ili promijeniti svojstvo elementa.

Korisne funkcije i svojstva:

- **document.getElementById (id)** – dohvaća jedan element po jedinstvenom identifikatoru
- **document.getElementsByTagName (name)** – dohvaća sve elemente određenog imena

(u sljedećim primjerima *element* označava već dohvaćeni element dokumenta)

- element.**innerHTML** – dohvaća ili mijenja vrijednost HTML-a koju neki element sadrži
- element.**setAttribute (name, value)** – dodaje atribut ili mijenja vrijednost atributa elementa
- element.**getAttribute (name)** – dohvaća vrijednost atributa elementa
- element.**style** – dohvaća ili mijenja vrijednost nekog dijela stila CSS-a
  - element.**style.textAlign** = 'center'
  - element.**style.color** = 'red'
  - element.**style.visibility** = 'hidden'
  - element.**style.visibility** = 'visible'
  - element.**style.display** = 'none'
  - element.**style.display** = 'block'

Pomoćni alati

Za rad s JavaScriptom i objektom *XMLHttpRequest* vrlo su korisni različiti alati poput *JavaScript debuggera*, koji su dostupni za razne preglednike, te programa/dodataka za praćenje prometa protokola HTTP.

Preporuka korisnicima Firefoxa je alat Tools -> **Error Console**, dodatak **DOM Inspector**, te dodatak **Firebug** koji sadrži mnoštvo mogućnosti za pregled strukture DOM-a, *JavaScript debugger*, te izvrstan pregled komunikacije klijenta i poslužitelja (i u slučaju *XMLHttpRequesta*).

Predaja vježbe

Rezultati vježbe se na poslužitelj predaju kao arhiva **svih datoteka** potrebnih za rad, složenih po strukturi direktorija (dakle, public_html, web-app...). Ne zaboravite uključiti i skripte biblioteka za *tooltipove.*

Nazivi novih datoteka su:

- *detalji.php*
- *detalji.js*

Ispitno gradivo vježbe

Ispitno gradivo uključuje sve navedeno u pripremi za vježbu, te detaljno razumijevanje napisanog rješenja i snalaženja u prepravcima istog.

Primjeri pitanja:

1. Objasnite način rada *XMLHttpRequest*a
2. Objasnite prednosti korištenja *XMLHttpRequest*a
3. Objasnite promjenu svojstava elemenata pomoću JavaScripta
4. Objasnite način dohvaćanja pojedinog elementa u HTML DOM stablu
5. Dinamički promijenite boju teksta nekog elementa.
6. Što je to DHTML?

Linkovi i literatura

XML.com: Very Dynamic Web Interfaces
http://www.xml.com/pub/a/2005/02/09/xml-http-request.html

The DOM and JavaScript - http://developer.mozilla.org/en/docs/The_DOM_and_JavaScript

Using the W3C DOM Level 1 Core -
http://developer.mozilla.org/en/docs/Using_the_W3C_DOM_Level_1_Core

HTML DOM Reference - http://www.w3schools.com/htmldom/dom_reference.asp

HTML DOM Style Object Reference -
http://www.w3schools.com/htmldom/dom_obj_style.asp