Programski Sistemi

Front-end: jQuery i AJAX

jQuery

- Popularna JavaScript biblioteka koja se lako koristi
- Prednosti upotrebe JavaScript biblioteka su brojne: ne brinemo o nekompatibilnostima čitača, stanju zahteva itd.
- Cela biblioteka nalazi se u jednom .js fajlu koji se može preuzeti sa adrese *http://www.jquery.com/*
- Može se preuzeti produkciona ili razvojna verzija; ako se biblioteka samo koristi, a ne razvijaju se dopunski moduli, dovoljna je minimizirana produkciona verzija.

jQuery

- U stranu se uključuje kao i bilo koji drugi .js fajl, dodavanjem u tag <script> u zaglavlju
- Omogućuje lako biranje elemenata, uz razdvajanje ponašanja dokumenta od njegove strukture, kao i lako usputno manipulisanje DOM elementima

```
<!doctype html>
<html>
<head>
<title>Adding jQuery</title>
<script type="text/javascript" src="jquery-1.7.2.min.js"></script>
</head>
<body>
</body>
</html>
```

jQuery: osnovna sintaksa

- Kada se jQuery biblioteka uključi u stranu, ona dodaje funkciju **jquery()**, čija je prečica **$()**
- Ova funkcija vraća objekat koji takođe ima funkcije, npr. $.ajax() ili $.each()
- Kao selektori mogu se koristiti i “ i ‘, pa je ekvivalentno pisati $("a") i $('a')

Sintaksa	Opis
$("a")	Svi tagovi <a> u dokumentu
$(document)	Ceo dokument, često se koristi za pristup funkciji ready() o kojoj će biti reči kasnije
$(“#elementID”)	Element čiji je ID elementID
$(“.className”)	Element(i) čija je klasa className

jQuery: $(document).ready()

- Funkcija **$(document).ready()** uklanja potrebu za korišćenjem događaja **load()**
- Uz **$(document).ready()**, svi elementi DOM modela su dostupni pre nego što se funkcija **ready()** izvrši
- Pristup elementima obavlja se pomoću selektora; način njihovog rada u biblioteci jQuery zasniva se na kombinaciji CSS i XPath selektora

Selektori

Opis	jQuery selektor
Link	$("#linkOne")
	$(".specialClass")
Svi tagovi u dokumentu	$('div')
Tagovi <a> unutar tagova	$("div a")
 Link 1 Link 2 	$("#leftNav a")
Prvi ili poslednji element <p> na strani	$("p:first"), $("p:last")
Drugi element <p> na strani	$("p")[1]
Parni ili neparni redovi tabele	$(“tr:even"), $(“tr:odd")

Custom selektori

```
<!DOCTYPE html>
<html>
  <head>
    <script src="http://code.jquery.com/jquery-1.7.1.js">
    </script>

    <script>

    $(function(){
       // Define custom filter by extending $.expr[":"]
       $.expr[":"].greenbg = function(element) {
         return $(element).css("background-color") === "green";
       };
       var n = $(":greenbg").length;
       console.log("There are " + n + " green divs");
    });
    </script>
  </head>
  <body>
    <div style="width:10; height:10; background-color:green;"></div>
    <div style="width:10; height:10; background-color:black;"></div>
    <div style="width:10; height:10; background-color:blue;"></div>
  </body>
</html>
```

Pristup elementima obrazaca

Selektor	Opis
:checkbox	Sva polja za potvrdu
:checked	Sva polja koja su potvrđena
:input	Svi kontrole u tagu form
:password	Svi input elementi tipa password
:radio	Sva radio dugmad
:reset	Sva dugmad tipa reset
:selected	Svi elementi koji su trenutno izabrani
:submit	Sva dugmad tipa submit
:text	Svi input elementi tipa text
:hidden	Svi skriveni elementi
:visible	Svi vidljivi elementi

jQuery funkcije

Selektor	Opis
css()	Postavljanje stila za selektor
each()	Iteracija kroz elemente kolekcije
parent()	Dohvatanje roditeljskog elementa
before()	Umetanje HTML-a pre izabranog elementa
after()	Umetanje HTML-a posle izabranog elementa
html()	Čita ili postavlja HTML sadržaj elemenata (ako se pozove sa argumentom koji je HTML)
text()	Čita ili postavlja HTML sadržaj elementa i njegove dece
addClass()	Dodaje CSS klasu izabranim elementima
removeClass()	Uklanja CSS klasu izabranim elementima
ready()	Zadaje funkciju koja će se izvršavati kada se DOM kompletno učita
type()	Određuje JavaScript tip objekta

JavaScript i jQuery

JAVASCRIPT

```
window.onload = function() {
    var elements = document.getElementsByTagName("div");
    for(var = 0; i < elements.length; i++){
        elements[i].style.color = green;
    }
});
```

JQUERY

```
$(function(){
    $("div").each(function(index, value){
        $( this ).css( "color" , "green" )
    });
});
```

```
var tr = document.getElementByTagName("tr");
for ( var = i; i < tr.length; i++ ) {
    if ( tr[i].class === 'highlighted' ) {
        tr[i].class = 'normal';
    }
}
```

```
$("tr.highlighted").removeClass("highlighted").addClass("normal");
```

Callback funkcije

- Callback funkcija je ona koja se prosleđuje kao argument drugoj funkciji, a izvršava nakon što se pozivajuća (roditeljska) funkcija izvrši
- JQuery često koristi callback funkcije, naročito za AJAX o kome će biti reči u nastavku
- Primer:

```
$('.percentage').each(function() {
    if ($(this).text() >= .5) {
        $(this).css('font-weight', 'bold');
    }
});
```

jQuery model događaja

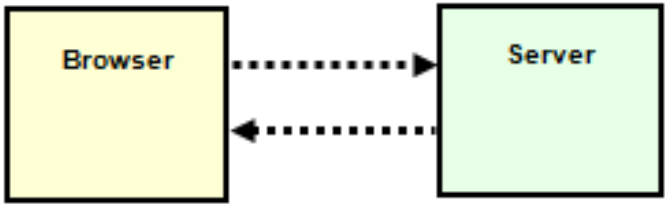
.blur()	.mouseenter()
.focus()	.mousemove()
.select()	.mouseout()
.submit()	.mouseover()
.click()	.mouseup()
.dblclick()	.toggle()
.focusin()	.error()
.focusout()	.resize()
.hover()	.scroll()
.mousedown()	

Šta je AJAX?

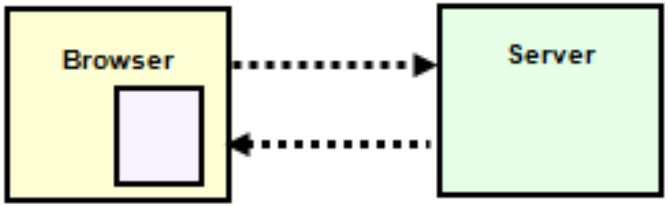
- Kao termin pojavio se 2003, skraćenica za Asynchronous JavaScript and XML
- AJAX je jedna od osnovnih tehnologija koja se koristi kao podrška za složene Internet Web 2.0 aplikacije; umesto formata XML, za razmenu podataka danas se često koristi format JSON
- Poboljšava interaktivnost na strani klijenta tako što omogućuje asinhrono osvežavanje delova strane
- AJAX je zapravo aplikacioni obrazac (application pattern) i podržavaju ga gotovo svi frejmvorci

AJAX

- Metod za korišćenje JavaScript i XMLHTTP ponašanja u okviru čitača da bi se omogućio dinamički kontekst bez osvežavanja stranice
- Većina čitača podržava slanje AJAX zahteva na server preko objekta XMLHTTPRequestObject



Kod tipičnih Web aplikacija, svaki zahtev dovodi do kompletnog osvežavanja stranice čitača



Ajax aplikacija počinje na isti način

Nakon inicijalnog učitavanja stranice, JavaScript kod dovlači dodatne podatke u pozadini, i osvežava određene odeljke stranice

AJAX i JQuery

- jQuery nudi nekoliko načina za slanje i primanje podataka serveru; to su funkcije **load()**, **post()** i **get()**
- Postoji i posebna funkcija **ajax()**; pomoću nje se može poslati nekoliko parametara, npr. tip HTTP metoda (GET or POST), time-out itd.
- Osnovna sintaksa je:
  $.ajax({ parametar: vrednost });
- Za proveru da li čitač podržava AJAX zahteve (tj. Kreiranje objekta XMLHttpRequest) može se koristiti uslov:
  if ($.support.ajax) { }

Parametri funkcije ajax()

parametar	Opis
url	URL resursa koji se zahteva
type	HTTP metoda Ajax zahteva (GET ili POST)
data	Podaci koji se šalju serveru
dataType	Tip podataka koji se očekuju kao odgovor od servera; može biti “XML”, “HTML”, “script”, “JSON” ili “text”
success(data, textStatus, jqXHR)	Funkcija koja se poziva ako zahtev uspe
error (jqXHR, textStatus, errorThrown)	Funkcija koja se poziva ako zahtev ne uspe
complete (jqXHR, textStatus)	Funkcija koja se poziva kada se zahtev obavi
timeout	Podešavanje perioda isteka zahteva (u milisekundama)

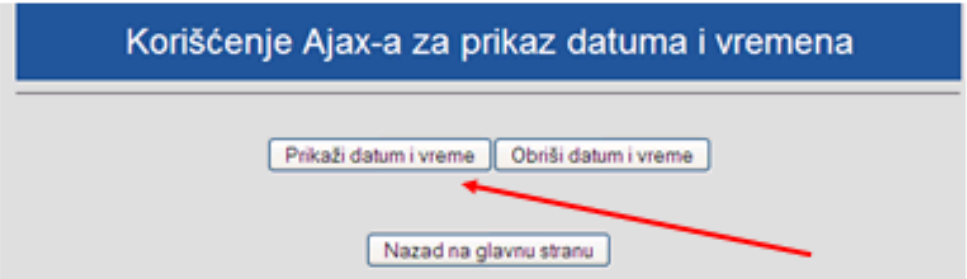
JSON

- JavaScript Object Notation
- Način za prosleđivanje podataka u obliku izvornih JavaScript objekata i nizova, umesto u XML formatu
- Za razliku od parsiranja XML-a koje je sporo, parsiranje JSON podataka se obavlja direktno u JavaScriptu; takođe, JSON je kompaktniji u odnosu na XML
- JSON string ima tačno određenu formu:
  { "propertyName1" : " value1 ", "propertyName2"": " value2 " }
- JSON string se pomocu funkcije **eval()** pretvara u JavaScript objekat

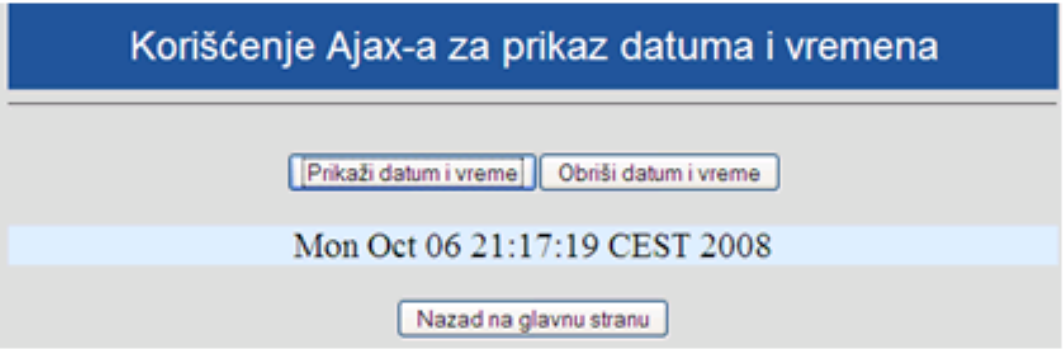
jQuery, AJAX i JSON

- jQuery ima i funkciju **getJSON()** koja radi isto što i druge AJAX funkcije, ali prihvata isključivo JSON podatke sa servera
- Funkcija **getJSON()** je ekvivalentna pozivanju funkcije **ajax()** sa dodatnim parametrom **dataType: 'json'**

AJAX: primer prikaza datuma i vremena



Kada korisnik pritisne dugme Prikaži datum i vreme osvežava se samo deo strane.



U div element se upisuje datum i vreme.Datum i vreme se čitaju sa servera.Osvežavanje dela strane omogućeno je jer je generisan asinhroni Ajax zahtev.

AJAX: osnovni koraci

- Kada se pritisne dugme *Prikaži datum i vreme* poziva se jQuery funkcija ajax()
- U ovoj funkciji se generiše objekat XMLHttpRequest
- XMLHttpRequest objekat šalje asinhroni AJAX zahtev serveru
- Kada server obradi zahtev i pošalje odgovor, poziva se callback funkcija; u ovoj funkciji se u DOM (tag div) upisuju datum i vreme.
- Za vreme obrade zahteva korisnik može da nastavi nesmetano da radi sa aplikacijom

Metoda servleta

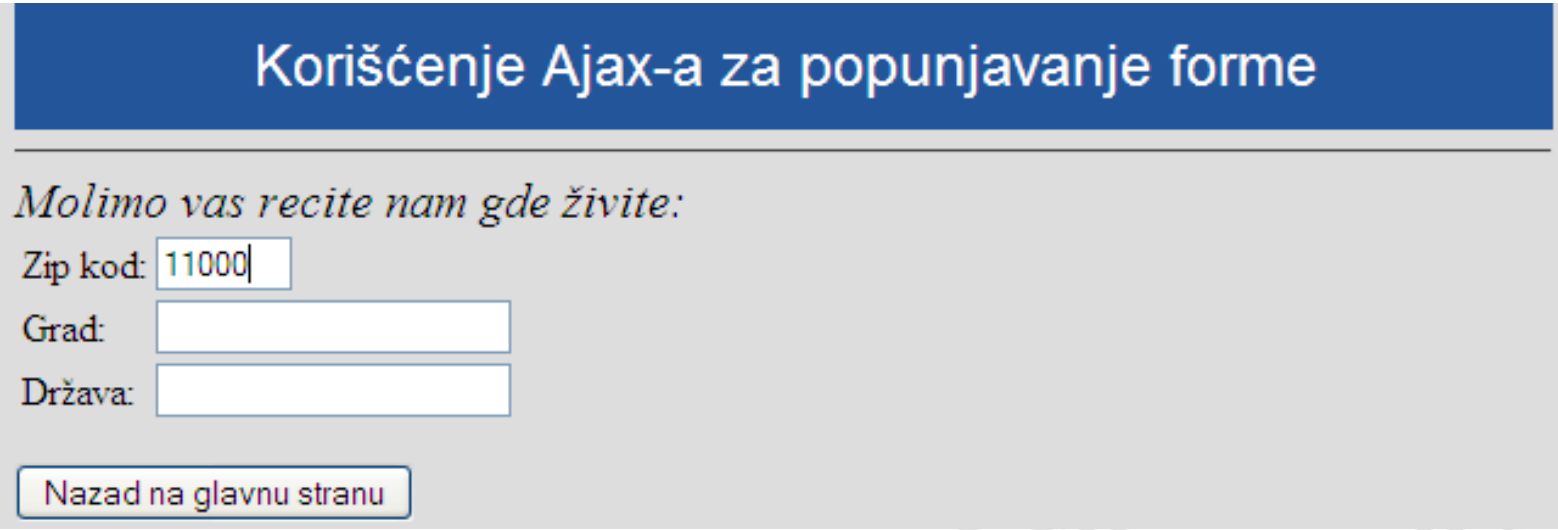
```
@WebServlet(name = "DateServlet", urlPatterns = {"/date"})
public class DateServlet extends HttpServlet {

    protected void processRequest(HttpServletRequest request, HttpServletResponse response)
            throws ServletException, IOException {
        response.setContentType("text/html;charset=UTF-8");
        PrintWriter out = response.getWriter();
        out.println(new Date().toString());
    }
}
```

jQuery AJAX kod

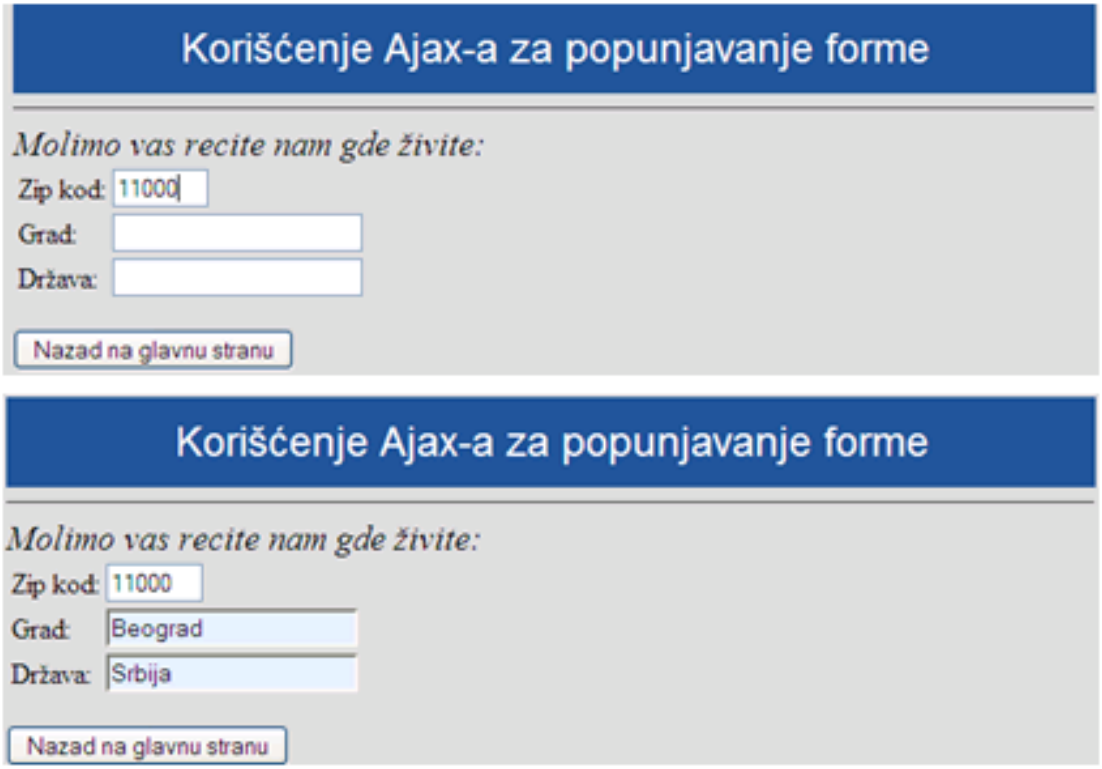
```
<head>
    <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
    <title>jQuery Ajax datum</title>
    <link href="css/ui-lightness/jquery-ui-1.10.4.custom.min.css" rel="stylesheet">
    <script src="js/jquery-1.10.2.js"></script>
</head>
<body>
    <script>
        $(document).ready(function() {
            $("#prikazi").click(function() {
                $.ajax({
                    type: "GET",
                    url: "${pageContext.request.contextPath}/date",
                    contentType: "html",
                    success: function(data) {
                        $("#vreme").text(data);
                    },
                    error: function(jqXHR, textStatus, errorThrown) {
                        alert("AJAX error: " + textStatus + ' : ' + errorThrown);
                    }
                });
            });
        });
        $(document).ready(function() {
            $("#obrisi").click(function() {
                $("#vreme").text("");
            });
        });
    </script>
    <input type="submit" value="Prikaži datum i vreme" id="prikazi">
    <input type="reset" value="Obriši datum i vreme" id="obrisi">
    <br>
    <div id="vreme"></div>
</body>
tml>
```

Primer: popunjavanje dela obrasca

- Ovaj primer prikazuje jednu od standardnih primena AJAX-a u kojoj korisnik popuni deo forme, a to uzrokuje ažuriranje drugih delova obrasca



Primer: popunjavanje dela obrasca



- Korisnik upiše zip kod.
- Kada polje izgubi fokus generiše se asinhroni zahtev.
- Kao parametar zahteva se šalje zip.
- Iz baze podataka se na osnovu zip koda čitaju grad i država.
- Generiše se odgovor klijentu.
- Osveže se samo polja grad i država

jQuery AJAX

```
<div id="states"></div>
        <form>
            <label>Poštanski broj:</label><input type="text" id="zip"><br>
            <label>Grad:</label><input type="text" id="city"><br>
            <label>Država:</label><input type="text" id="state"><br>
        </form>
 <script>
            $(function() {
                $('#zip').focusout(function(){
                    $.ajax({
                        type: "GET",
                        url: "${pageContext.request.contextPath}/complete",
                        dataType: "json",
                        data: {"zip": $('#zip').val()},
                        success: function(data) {
                          $("#city").val(data.city);
                          $("#state").val(data.state);
                        },

                        error: function(jqXHR, textStatus, errorThrown) {
                                alert("AJAX error: " + errorThrown);
                        }
                    });
                });
            });
        </script>
    </body>
</html>
```

Serverski kod

- Odgovor se šalje sa servera u JSON notaciji
- JSON string ima dva svojstva: city i state.

```
protected void processRequest(HttpServletRequest request, HttpServletResponse response)
        throws ServletException, IOException {
    response.setContentType("application/json");
    String zip = request.getParameter("zip");
    String json="";
    if(zip.equals("11000"))
        json="{\"city\": \"Beograd\", \"state\": \"Srbija\"}";
    else if(zip.equals("21000"))
        json="{\"city\": \"Novi Sad\", \"state\": \"Srbija\"}";
    else
        json="{\"city\": \"nema podataka\", \"state\": \"nema podataka\"}";
    response.getWriter().write(json);
}
```