Visoka tehnička škola Niš

Studijski program:

Savremene računarske tehnologije

Napredne Web tehnologije - NWT

(9)

RIA i AJAX tehnologija

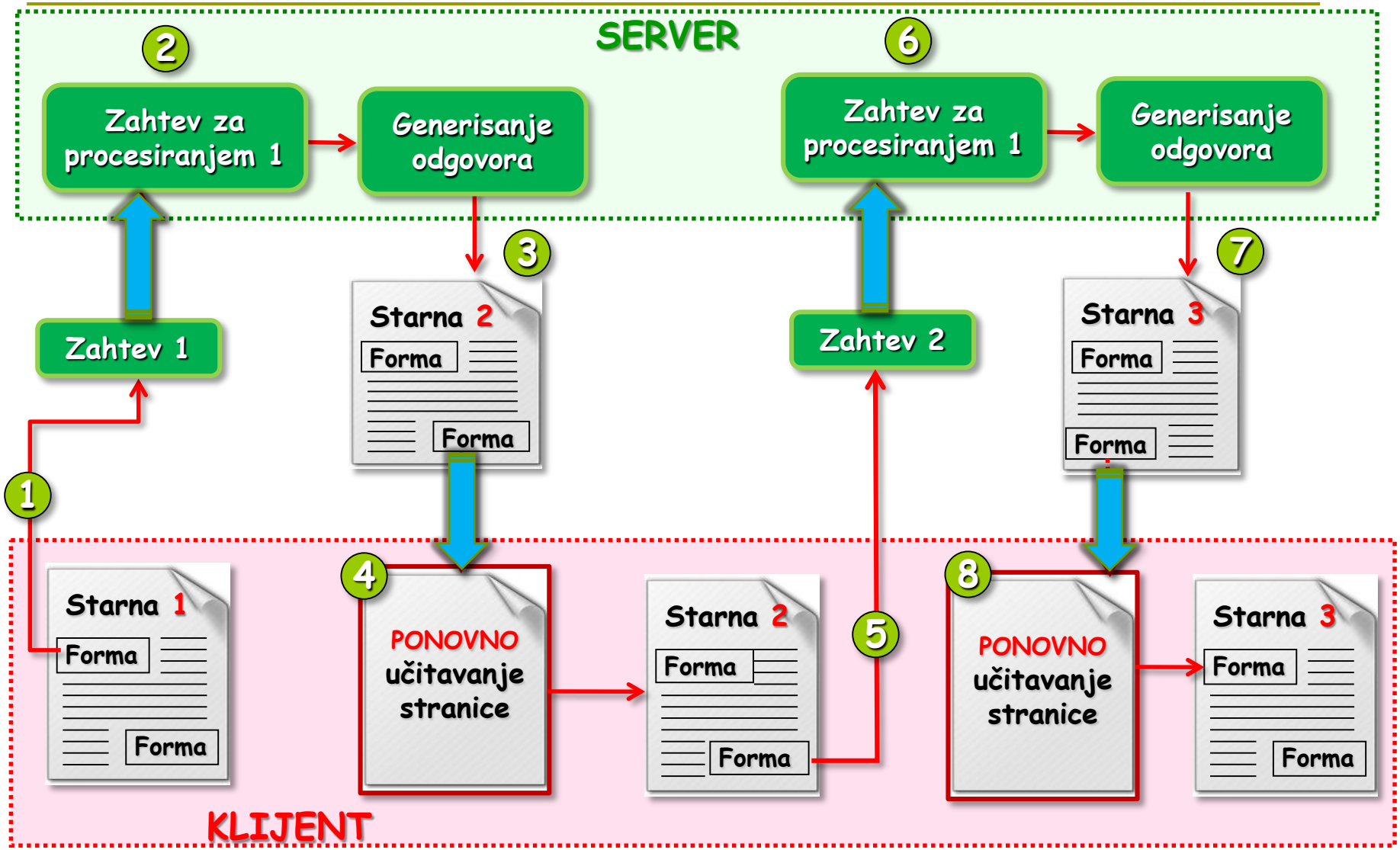
Prof. dr Zoran Veličković, dipl. inž. el.

Mart, 2019.

Savremene Web aplikacije

- Standardi i **KORISNIČKI DOŽIVLJAJ** koje su postavile desktop aplikacije su gotovo **NEDOSTIŽNI WEB APLIKACIJAMA**.
- Razlog ovome leži u činjenici da svaka komunikacija Web aplikacije sa (svojim) delom na serveru zahteva **ZNATNO VIŠE VREMENA** nego desktop aplikac.
- Razvoj savremenih Internet tehnologija je doprineo da **RAZLIKE** između desktop i web aplikacija **POSTAJU SVE MANJE**.
- Web aplikacije koje korisniku pružaju **UDOBNOST DESKTOP APLIKACIJA** nazivaju se **RIA** (engl. Rich Internet Applications) aplikacije
- Osnovno pitanje za Web dizajnere je kako ostvariti ovaj zahtev?
- Dakle, **RIA** aplikacije moraju obezbediti **VISOKE PERFORMANSE** i **BOGATI GRAFIČKI INTERFEJS**.
- Svoje visoke performanse **RIA** aplikacije uglavnom mogu zahvaliti **SKRIPTOVANJU INTERAKCIJE SA KLIJENTOM** kako bi se dobilo na dinamici (brzini odziva) Web aplikacije.

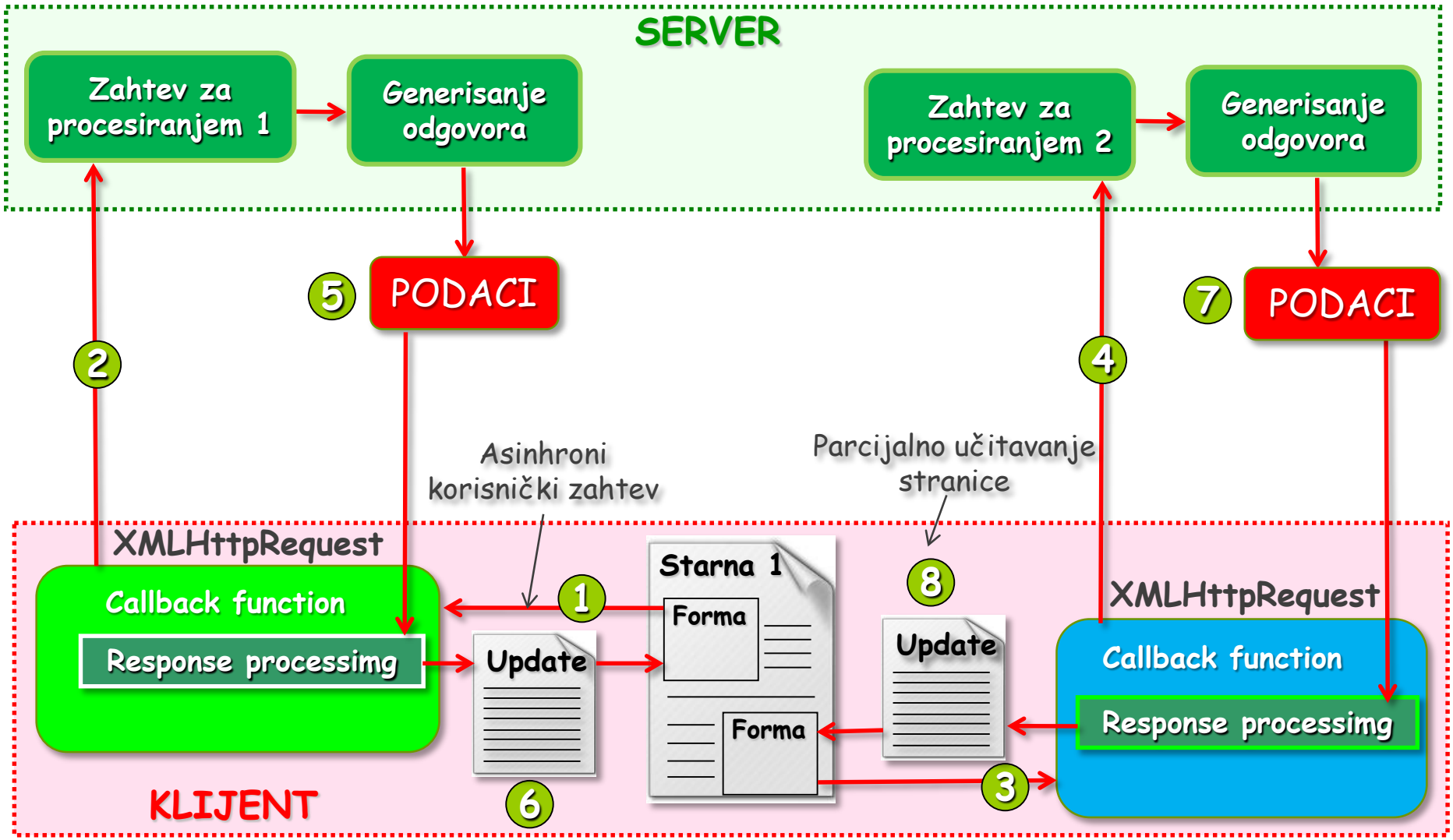
Tradicionalna Web aplikacija



AJAX

- **AJAX** (engl. Asinhroni JavaScript i XML), tehnologija podrazumeva **jasnu podelu Web aplikacije** na **KLIJENTSKI** i **SERVERSKI** deo.
- Osnovna ideja kod ovog koncepta je da se ovi **DELOVI JEDNE APLIKACIJE IZVRŠAVAJU PARALELNO** (**istovremeno**) na klijentskoj i serverskoj strani.
- Vreme koje bi se potrošilo na **SLANJE I ČEKANJE** odgovora od servera se u ovom knceptu **ZNATNO SMANJUJE**.
- Osnovna implementacija **AJAX**-a zasniva se na primeni **JavaScript METODE** za **ASINHRONO SLANJE ZAHTEVA** prema serveru.
- **ODGOVORI** na asinhrone zahteve poslate serveru se **AŽURIRAJU NA WEB STRANICI** korišćenjem njene **DOM** (engl. Document Object Model) strukture.
- **AJAX komponente** imaju zadatak da **ASINHRONO AŽURIRAJU DEO STRANE** koji im je poveren.
- Za bolju **prenosivost** i **rad sa AJAX komponentama** razvijeni su posebni programski paketi: **Dojo**, **Prototype**, **ASP.NET Ajax**, ...

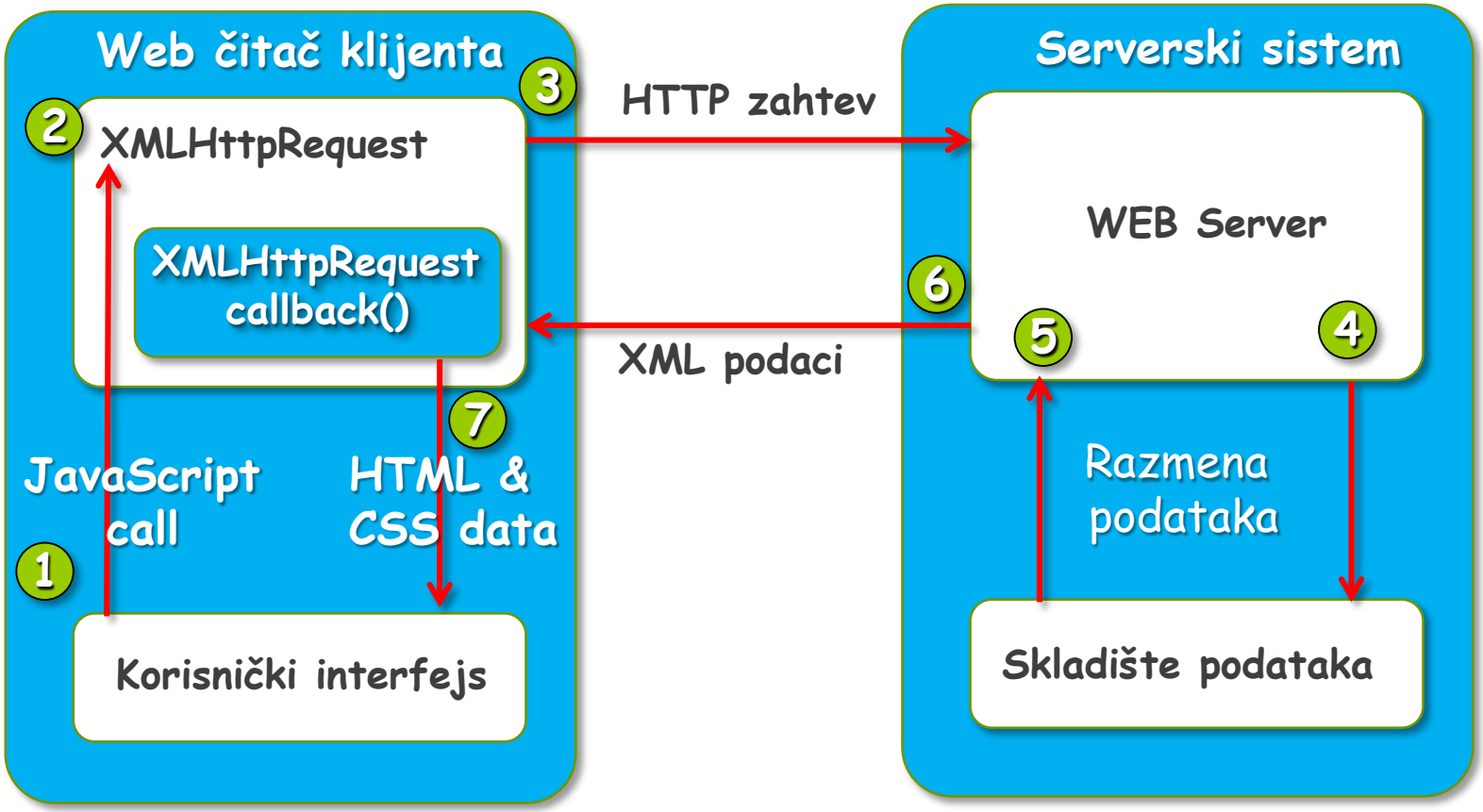
AJAX forma Web aplikacije



Client-server tehnologije i AJAX

- Za razvoj **RIA aplikacija** uz podršku **AJAX**-a moge se koristiti **razvojna okruženja** za različite platforme: Adobe Flex, Microsoft Silverlight i JavaServer Faces, …
- Na **KLIJENTSKOJ STRANI** AJAX aplikacija koristi: **XHTML**, **CSS** i **JavaScript** za podršku korisničkom interfejsu.
- **XML** se koristi za **STRUKTURNU RAZMENU PODATAKA** između **SERVERA** i **KLIJENTA**.
- **JSON** (engl. JavaScript Object Notation) se takođe može koristi koristi za za ovu svrhu.
- JavaScript objekt **XMLHttpRequest** se koristi za **UPRAVLJANJE INTERAKCIJOM** sa serverom.
- **OBRADA PODATAKA** na **SERVRSKOJ STRANI** može biti implementirana različitim serverskim tehnologijama: **PHP**, **ASP. NET**, JavaServer Faces i Rubi on Rails.

XMLHttpRequest



AJAX Level 2

- U HTML 5 se primenjuje novi API **XMLHttpRequest** (nivo 2) za izvršavanje AJAX-a.
- Level 2 podržava komunikaciju iz **VIŠE IZVORA**, kao i nove događaje.
- Skriptovi su pojednostavljeni i nude se nove opcije kao što je **INTERAKCIJA SA VIŠE SERVERA** iz jedne aplikacije.
- Najvažniji element ovog API-ja je **OBJEKT XMLHttpRequest** za čije instanciranje se koristi **konstruktor** XMLHttpRequest().
- Ovaj konstruktor vraća **OBJEK**T tipa **XMLHttpRequest** pomoću koga se:
  - **Pokreću zahtevi**
  - **Osluškuju događaji kojima se upravlja proces AJAX komunikacije**
- Kao i većina objekta u OO programiranju objekt tipa **XMLHttpRequest** poseduje **METODE** za pokretanje i kontrolu zahteva:

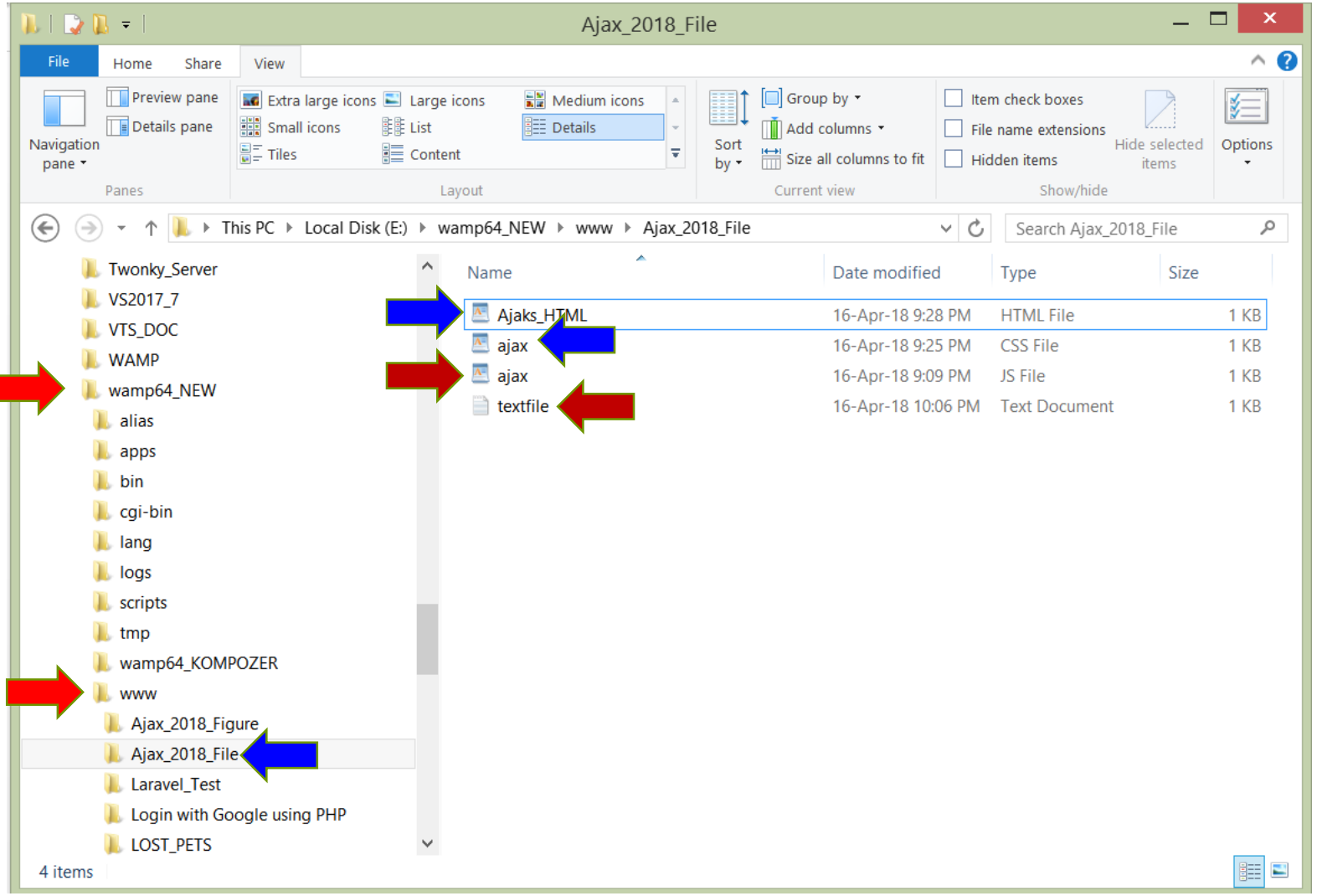
Metode objekta XMLHttpRequest

Metoda	OPIS
open(metoda, url, asin)	• Konfiguriše zahtev na čekanju. • Atribut metoda zadaje HTTP metodu za realizaciju konekcije. • Može uzeti vrednost GET ili SET. • Atribut url deklariše lokaciju skripta koji će obrađivati zahtev. • Atribut asin je logičkog tipa i definiše sinhronu (false) ili asinhronu komunikaciju (true).
send(podaci)	• Ova metoda pokreće zahtev. • Postoje nekoliko preklopljenih metoda send. • Atribut podaci se može izostaviti ili deklarisati kao ArrayBiffer, BLOB, dokument, znakovni niz ili objekt FormData.
abort()	• Otkazivanje zahteva.

Svojstva objekta XMLHttpRequest

Svojstvo	OPIS
response	• Svojstvo opšte namene. • Vraća odgovor u formi definisanoj drugim svojstvom responseType. • Svojsto responseType je deo objekta XMLHttprequest i mora se deklarisati pre upućivanja zahteva. • Može uzeti sledeće vrednosti: text, arraybuffer, blob, document i json.
responseText	• Vraća odgovor na zahtev u formi neformatiranog teksta.
responseXML	• Vraća odgovor na zahtev u formi XML podatka.
status	• Http status koda zahteva vraća sledeće numeričke vrednosti: • 200: "OK" • 403: "Forbidden" • 404: "Not Found"
readyState	• Čuva tok tahteva, 0-zahtev nije inicijalizovan, 1-zahtev se obrađuje, 2-zahtev je pristigao, 3-podaci poslati sa servera, 4-zahtev kompletiran

Ajax: čitanje .txt fajla na localhost-u



Informacije iz tekstualne datoteke

Dobijanje informacije iz tekstualne datoteke na serveru pomoću GET metode

Ajax_HTML.html

```
<!DOCTYPE html>
<html lang="en">
    <head>
        <title>Ajax Level 2</title>
        <link rel = "stylesheet" href = "ajax.css">
        <script src = "ajax.js"> </script>
    </head>
<body>
    <section id = "formbox">
      <form name = "form">
           <p> <input type = "button" name = "button" id = "button"
                      value = "Učitaj fajl sa sarvera"> </p>
      </form>
    </section>
 <section id = "databox"> </section>
</body> </html>
```

Stilizacija izgleda

Ajax.css

```
#formbox {
    float: left;
    padding: 20px;
    border: 1px solid #999999;
    background: #00ff88;
}
#databox {
    float: left;
    width: 500px;
    margin-left: 20px;
    padding: 20px;
    border: 1px solid #999999
    background: #999933;
    color: #ffffff;
    font-family: Comic Sans MS;
}
```

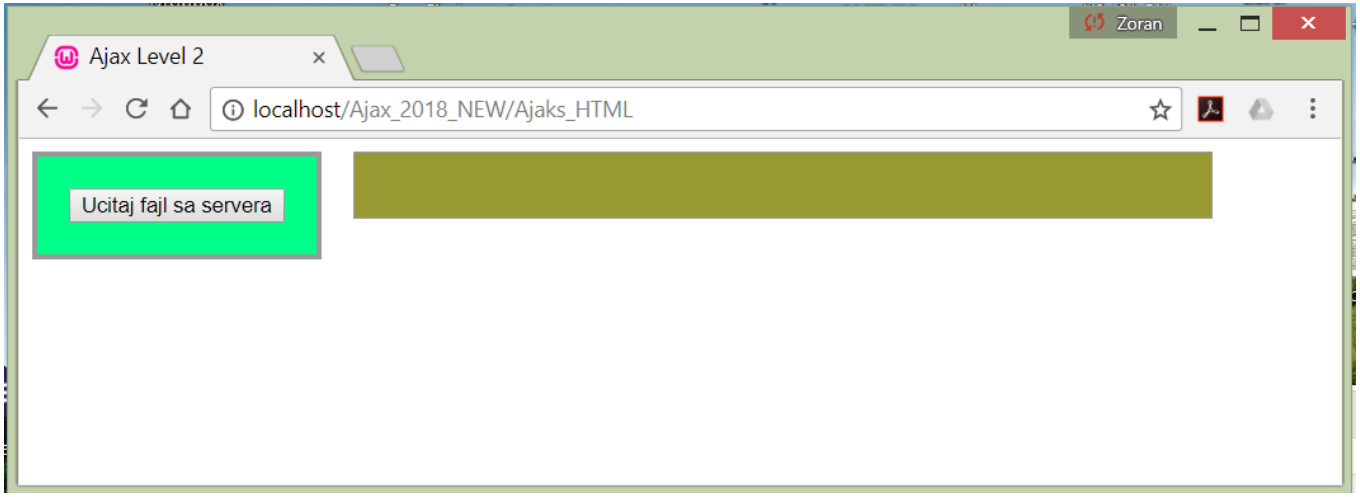
Učitavanje datoteke

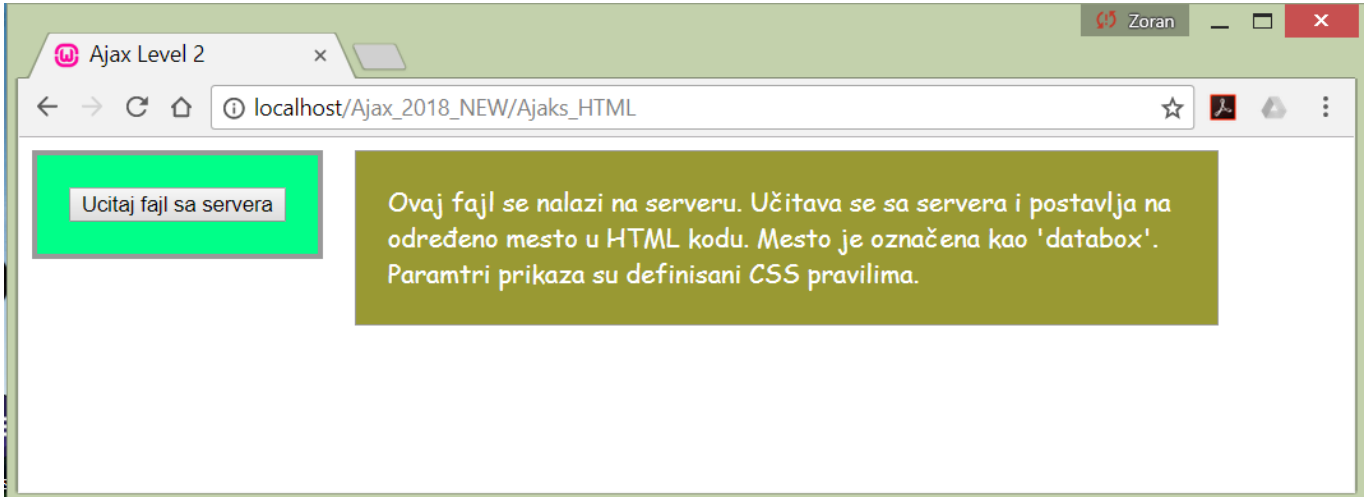
Ajax.js

```
function initiate() {
   var databox = document.getElementById('databox');
   var button = document.getElementById('button');
   button.addEventListener('click', read, false);
}
function read() {
   var url = "textfile.txt";
   var request = new XMLHttpRequest();
   request.addEventListener('load', show, false);
   request.open("GET", url, true);
   request.send(null);
}
function show(e) {
    var data = e.target;
    if(data.status == 200) {
       databox.innerHTML = data.responseText;
    }
}
addEventListener('load', initiate, false);
```

Metoda **send()** je prazna ali njene atribute koristi metoda **open()**

Primer: Učitavanje datoteke





NAPOMENA:
Da bi ovaj primer bio funkcionalan treba obezbediti pristup serveru na kome se nalazi fajl za čitanje. Pokrenite zahtev iz Wamp-a ili postavite datoteke na NWT serveru.

Ajax: čitanje .txt fajla sa NWT-a

- Realizovati prethodni primer na NWT serveru.
- Podeliti poslove studentskim grupama:
  - Postavljanje .txt fajla
  - Postavljanje .html fajla
  - Postavljanje .css fajla
  - Postavljanje .js fajla

Očitavanje slike sa servera

```
function initiate() {
 var databox = document.getElementById('databox');
 var boton = document.getElementById('button');
 button.addEventListener('click', read, false);
}

function read() {
   var url = "myimage.jpg";
   var request = new XMLHttpRequest();
   request.responseType = 'blob';
   request.addEventListener('load', show, false);
   request.open("GET", url, true);
   request.send(null);
}
```

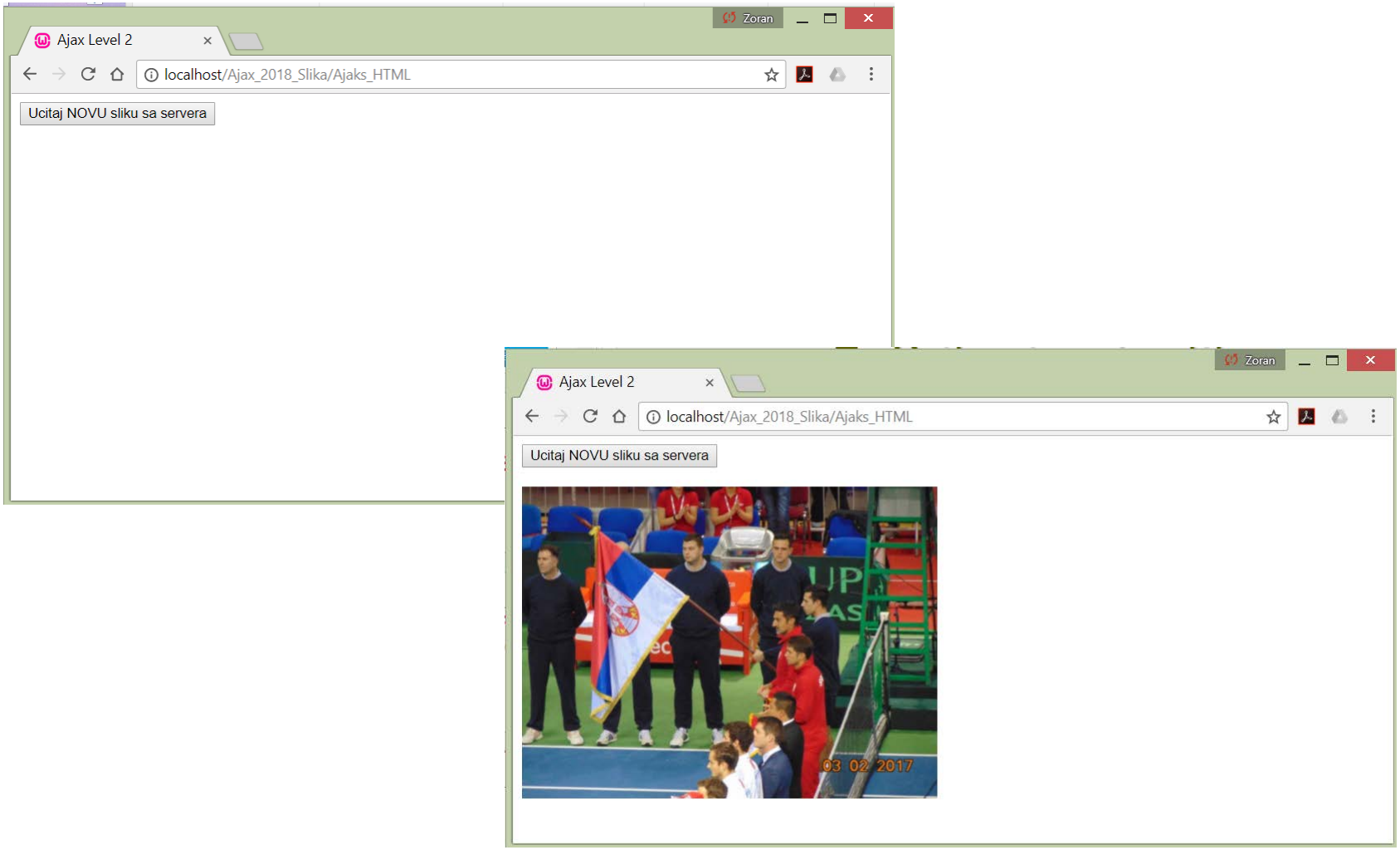
Svojstvo **responseType** za zahtev zadaje sa vrednošću '**blob**'.

Prikazivanje slike

Prikazuje se pruzeta slika u prozoru Web čitača

```
function show(e) {
var data = e.target;
if(data.status == 200) {
    var image = URL.createObjectURL(data.response);
    databox.innerHTML = '<img src="' + image + '"width = "400"
                                                height = "300">';
   }
}
addEventListener('load', initiate, false);
```

Primer: Učitavanje slike



Događaji

- Pomoću događaja može se napraviti profesionalna aplikacija.
- Događajem **PROGRES** može se obaveštavati korisnik osvakom koraku vezanu za komunikaciju sa serverom.
- Evo nekoliko korisnih svojstava.

Svojstvo	OPIS
loadstart	• Aktivira se kada se zahtev pokrene.
progress	• Aktivira se periodično i pri slanju i pri učitavanju.
abort	• Aktivira se ako se zahtev obustavi.
error	• Aktivira se na pojavu greške
load	• Aktivira se kada se zahtev završi.
timeout	• Aktivira se ako se zahtev ne može izvršiti u zadatom vremenu.
loadend	• Aktivira se kada se zahtev završi.

Slanje podataka AJAX-om (1)

- Već znamo da se slanje podataka može realizovati metodama **GET** i **POST**.
- Ako se podaci šalju metodom **GET** , vrednosti se **DODAJU NA URL ADRESU**.
- **POST** metod je nešto složenija, ali daje iste rezultate.
- Podaci se mogu poslati **KREIRANJEM OBRASCA**, ali AJAX pruža i druge mogućnosti.
- Za ove potrebe formiran je interfejs **DataForm** sa svojim **konstruktorom** i **metodama**.
  - **FormData(obrazac)** - **Konstruktor**, vraća istoimeni objekt koji metoda send() koristi za slanje podataka.
  - **append(ime, vrednost)** – Dodaje podatke u objekt FormData. Atribut vrednost može biti znakovni niz ili BLOB. Vraćeni podatak predstavlja polje obrasca.

Slanje podataka AJAX-om (2)

```
function initiate() {
   var databox = document.getElementById('databox');
   var button = document.getElementById('button');
   button.addEventListener('click', send, false);
function send() {
  var data = new FormData();
  data.append('name','John');
  data.append(lastname',’Smith');
   var url = "process.php";
   var request = new XMLHttpRequest();
   request.addEventListener('load', show, false);
   request.open("POST", url, true);
   request.send(data);
}
function show(e) {
   var data = e.target;
       if(data.statusa == 200) {
          databox.innerHTML = data.responseText;
} addEventListener('load', iniciar, false);
```

Šalje podatke u datoteku process.php

Učitavanje datoteke na server

- Učitavanje datoteke na server je veoma važana aktivnost na Web stranici.
- Upload - Svojstvo koje vraća objekt **XMLHttpRequestUpload**, a mora se pozvati iz objekta XMLHttpRequest.
- Dodajemo novo <input> polje.

Učitavanje HTML

```
<!DOCTYPE html>
<html lang="en">
  <head>
     <title>Ajax Level 2</title>
     <link rel="stylesheet" href="ajax.css">
     <script src="ajax.js"></script>
  </head>
<body>
<section id="formbox">
   <form name="form">
     <label for = "myfiles"> Filt ti Upload: </label>
     <input type = "file" name = "myfiles" id = "myfiles">
   </form>
</section>
   <section id = "databox"> </section>
  </body>
</html>
```

Učitavanje datoteke na server (1)

```
function initiate() {
   var databox = document.getElementById(’databox');
   var button = document.getElementById(myfules');
   button.addEventListener('change', upoload, false);
}
function upload(e) {
var files = e.target.files;
var mxfile = files[0];
var data = new FormData();
data.append(’file', myfile);
var url = "process.php";
var request = new XMLHttpRequest();
request.addEventListener('loadstart', start, false);
request.addEventListener('load', show, false);
```

Učitavanje datoteke na server (2)

```
var xmlupload = request.upload;
Xmlupload.addEventListener(progres, ststus)
request.open("POST", url, true);
request.send(datos);
}
function start() {
   databox.innerHTML = '<progress value = "0" max = "100„ > 0% </progress>';
}
function status(e) {
   if(e.lengthComputable) {
      var per = parseInt(e.loaded/e.total*100);
    var progressbar = databox.querySelector("progress");
     progressbar.value = per;
     progressbar.innerHTML = per+'%';
   }  }
```

Učitavanje datoteke na server (3)

```
function show(e) {
var data = e.target;
   if(data.status == 200) {
   databox.innerHTML = 'Done';
   }
}
addEventListener('load', initiate, false);
```