

WEB PROGRAMIRANJE

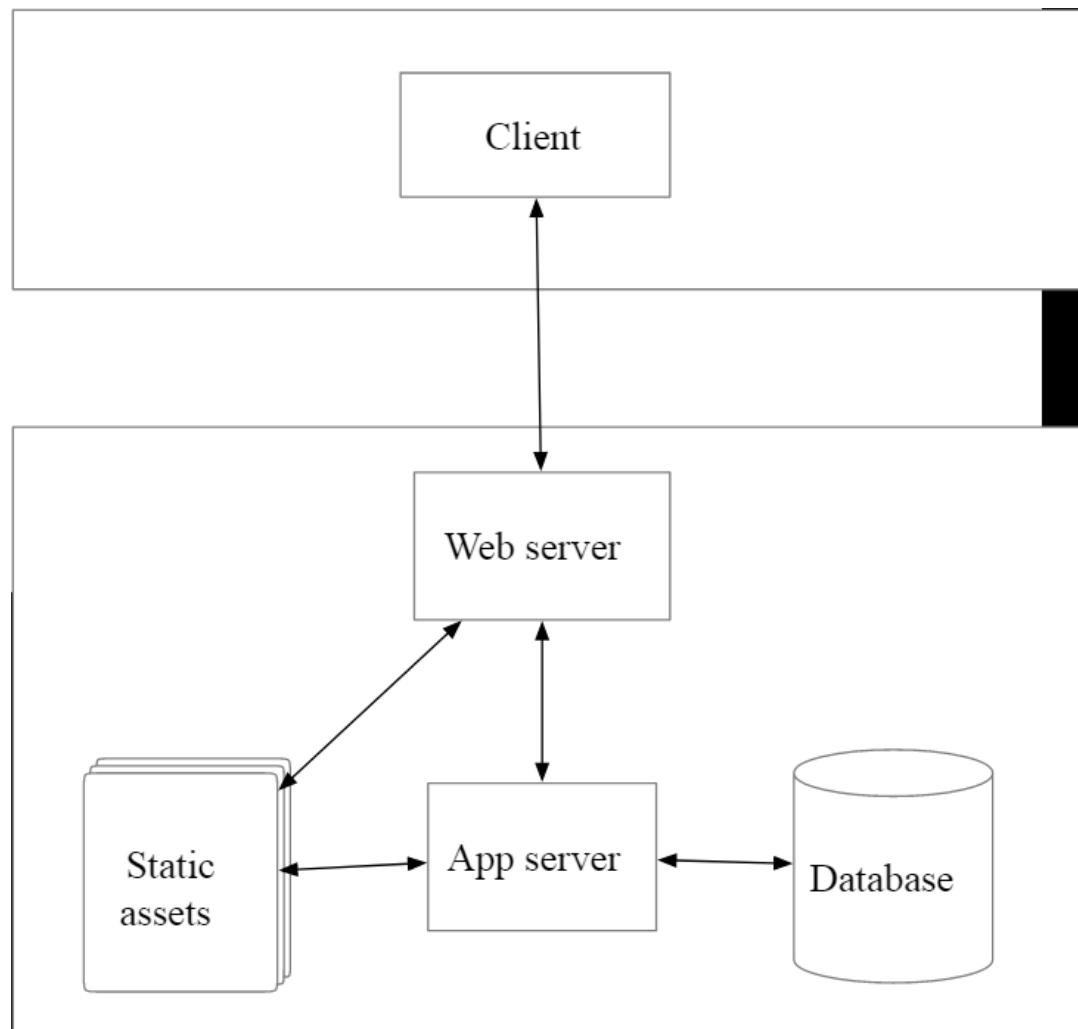
Markup jezici

Jezici i tehnologije za programiranje klijentske strane

Jezici i tehnologije za programiranje serverske strane

Web servisi

Struktura web aplikacije



Jezici i alati za web programiranje

- Jezici za definisanje sadržaja i izgleda statičkih web strana (*markup* jezici)
 - Sadržaj i struktura web strana i osnovno formatiranje: HTML
 - Napredno formatiranje prikaza: CSS
 - Za opis podataka: XML, DTD and XML-Schema
- Programiranje interaktivnih web strana na strani klijenta (*Client-side programming*)
 - Aplikacije koje se pozivaju iz web stranica: Java apleti, ActiveX, Flash
 - Skript jezici: JavaScript, VBScript
 - Okruženja za programiranje klijentske strane (frameworks): AngularJS, ReactJS

Jezici i alati za web programiranje

- Jezici i alati za programiranje kreiranja Web strana na strani servera (*Server-side programming*)
 - Tehnologije koje koriste spoljne programe za generisanje HTML strana: CGI, Java Servleti
 - Skript jezici: PHP, Ruby, Python, Node.js/JavaScript
 - Okruženja za programiranje serverske strane (frameworks): JSP, ASP, ASP.NET , Laravel (PHP), Rails)(Ruby), Django (Python), Express (Node.js/JavaScript)
- Jezici za opis web servisa:
 - WSDL, SOAP

MARKUP JEZICI

HTML

- HyperText Markup Language
- Osnovni jezik za predstavljanje dokumenata na Webu
 - Prenosi se korišćenjem HyperText Transfer Protocol-a (http)
 - Klijent šalje zahtev serveru u obliku stringa
 - Server vraća dokument
- Opisuje sadržaj i struktru dokumenta
 - Osnovna formatiranja potrebana za prikaz se mogu ugraditi u sam HTML, dok se preciznija formatiranja definišu posebno u .css fajlu
- Web Browser čita HTML dokument i na osnovu formatiranja iz .css fajla kreira prikaz

HTML

- HTML fajl opisuje dokument korišćenjem HTML tagova
- HTML tagovi su ugnježdeni jedan u drugi i čine strukturu stabla
- Format HTML tagova:

```
<SPOLJAŠNJITAG attribute1='val1' attribute2='val2'>  
  <UNUTRAŠLJITAG attribute3='val3'>  
    neki tekst  
  </UNUTRAŠNJITAG>  
</SPOLJAŠNJITAG>
```

Struktura HTML dokumenta

```
<html>  
<head>  
<title>...</title>  
</head>  
<body>  
...  
</body>  
</html>
```


Neki bitniji HTML tagovi

- Definicija naslova, podnaslova...
 - `<h1> Glavni naslov </h1>`
 - `<h2> Podnaslov </h2>`
- Definicija pasusa (paragrafa)
 - `<p> Pasus </p>`
- Definicija hiperlinka
 - ``
tekst koji služi kao veza (link)
``

Neki bitniji HTML tagovi

- Umetanje slike

```

```

- Definicija tabele

```
<table border="3">
```

- *def. tabele*

```
<tr align="center">
```

- *def. vrste*

```
<td>Naslov</td>
```

- *def. ćelije*

```
<td>Autori</td>
```

```
<td>Izdavač</td>
```

```
</tr>
```

```
<tr><td>HTML: The Definitive Guide</td>
```

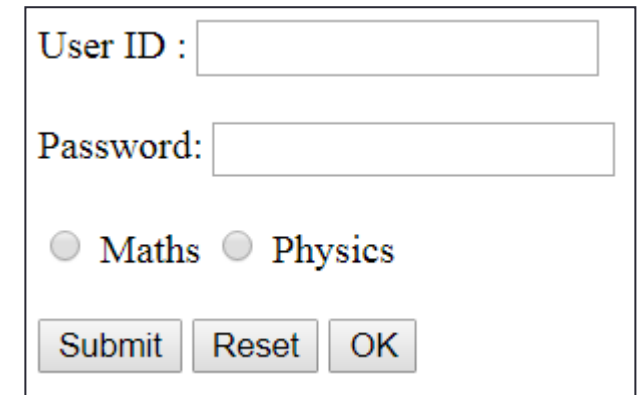
```
<td>Chuck Musciano and Bill Kennedy</td>
```

```
<td>O'Reilly & Associates</td>
```

```
</tr>
```

Neki bitniji HTML tagovi

- Kreiranje forme:



User ID :

Password:

☐ Maths ☐ Physics

`<form>`

User ID : `<input type = "text" name = "user_id" />`

`

`

Password: `<input type = "password" name = "password" />`

`

`

`<input type = "radio" name = "subject" value = "maths">` Maths

`<input type = "radio" name = "subject" value = "physics">` Physics

`

`

`<input type = "submit" name = "submit" value = "Submit" />`

`<input type = "reset" name = "reset" value = "Reset" />`

`<input type = "button" name = "ok" value = "OK" />`

`</form>`

CSS

- **C**ascading **S**tyle **S**heet
 - Skup pravila pomoću koji se definiše raspored i izgled stranice
- CSS omogućava pisanje naredbi za izgled i formatiranje u zaglavlju Web stranice ili u spoljašnjoj datoteci, van HTML koda kojim se zadaje sadržaj stranice
- Stranica na kojoj su razdvojeni sadržaj i izgled lakše se održava i ažurira

Povezivanje HTML-a i CSS-a

- CSS ugradjen u sam HTML:

```
<HEAD>  
  <STYLE type="text/css">  
    <!-- ...CSS DEFINITIONS.. -->  
  </STYLE>  
</HEAD>
```

- CSS u posebnom fajlu:

```
<HEAD>  
  <LINK rel="stylesheet" type="text/css"  
    href="fluorescent.css" />  
</HEAD>
```

CSS definicije

- Izgled HTML elementa se u css-u definiše na sledeći način:

```
ImeElementa {  
    ImeAtributa1 : vrednost1;  
    ImeAtributa2 : vrednost2;  
    ...  
}
```

- Primer:

```
h1 {color:red; font-size: 16px}
```

XML

- E**X**tensible **M**arkup **L**anguage
- Ima istu strukturu kao HTML, jedino nema predefinisani skup tagova – tagovi se mogu definisati potpuno proizvoljno
- Služi za predstavljanje podataka
 - Uglavnom, za razmenu podataka između aplikacija koje se izvršavaju na različitim platformama

XML dokument

- Sastoji se od tagova i podataka
- Tagovi čine skturu stabla:
 - Tagovi se mogu ugnježdavati
 - Postoji jedan koreni tag
 - Podaci su u terminalnim čvorovima stabla
- Primer:
 - Tagovi se mogu ugnježdavati

```
<?xml version="1.0"?>
<osoba>
    <ime>Petar Petrovic</name>
    <tel tip="kuca">412-555-4444</tel>
    <tel tip="posao">412-268-5555</tel>
    <email>petar.petrovic@gmail.com</email>
</osoba>
```


Tehnologije za obradu XML dokumenata

- DOM (**D**ocument **O**bject **M**odel)
 - Učitava ceo dokument u memoriju i čuva ga u obliku stabla.
 - Prednosti:
 - Laka manipulacija: omogućava traženje, dodavanje, brisanje elemenata
 - Nedostaci:
 - Veliki utrošak memorije
 - Učitavanje (parsiranje) dokumenta znatno sporije
- SAX (**S**imple **A**PI for **X**ML)
 - Baziran na *event-trigger-and-handling* principu
 - Sax parser čita dokument i generiše događaje tipa:
`startDocument`, `endDocument`, `startElement`,
`endElement`
 - Korisnik treba da programira handler-e za ovakve događaje
 - Prednosti:
 - Veoma brz, ne troši puno memorije
 - Nedostaci:
 - Služi samo za čitanje XML dokumenata

DTD i XML Schema

- DTD (**D**ocument **T**ype **D**escriptor) i XML Schema služe za definisanje strukture XML dokumenta

Definicija web strane

- HTML:
 - Strana sadrži:
 - Menije A, B, B
 - Naslov XXXX
 - Tabelu sa podacima o studentima
 - ...
- CSS:
 - Meni će biti prikazan uz desnu ivicu, boja pozadine menija je plava, velicina fonta 12...
 - Naslov ce biti na sredini, velicina fonta 20, boja siva...
 - ...
- XML:
 - Student:
 - Ime: Marko, prezime: Petrovic, ocena: 10
 - Ime: Petar, prezime: Peric, ocena: 8
 -

PROGRAMIRANJE KLIJENTSKE STRANE WEB APLIKACIJA

Zadaci klijentske strane u web aplikacijama

- Prihvatanje i provera korisničkih unosa
- Modifikacija sadržaja stranice u skladu sa korisničkim zahtevima bez komunikacije sa serverom kad-god je to moguće

Java apleti

- Java apleti su kratki programi koji se izvršavaju na web stranici.
- Prve dinamičke web stranice su prevljene pomoću apleta.
- Danas se apleti koriste za kreiranje virtuelnih laboratorija...
- Ugradnja apleta u web stranicu se vrši pomoću taga `<applet>`.
- Opšti izgled ovog taga:

```
<applet atributi >  
  [parametri]  
  [alternativni sadržaj]  
</applet>
```

Prenos apleta od servera do klijenta

- ▶ Prenosi se byte-code apleta, odnosno `.class` fajl.

Polja u `applet` tagu

- ▶ Polje sa atributima sadrži:
 - ▶ ime .class fajla (apleta),
 - ▶ dimenzije apleta na web stranici (širinu i visinu),
 - ▶ opciono put do .class fajla (ako nije u istom folderu kao stranica),
 - ▶ ...
- ▶ Polje sa parametrima sadrži argumente koje prosleđujemo apletu pri pokretanju. To su ulazni podaci za aplet.
- ▶ Polje za alternativni sadržaj služi za pretraživače koji nemaju podršku za Javu – tu može da piše neko upozorenje:
 - ▶ "Vaš pretraživač ne podržava Javu." ili
 - ▶ "Proverite da li ste instalirali Java plugin u svoj pretraživač".

Sintaksa **applet** taga

<APPLET

CODEBASE = *codebaseURL*

ARCHIVE = *archiveList*

CODE = *appletFile* ...or... **OBJECT** = *serializedApplet*

ALT = *alternateText*

NAME = *appletInstanceName*

WIDTH = *pixels* **HEIGHT** = *pixels*

ALIGN = *alignment* **VSPACE** = *pixels* **HSPACE** = *pixels* >

<PARAM NAME = *appletAttribute1* **VALUE** = *value*/>

<PARAM NAME = *appletAttribute2* **VALUE** = *value*/>

... *alternateHTML*

</APPLET>

- Pozvani aplet preuzima parametre iz taga pomoću metoda:

```
public String getParameter (String ime_parametra)
```

Dodatni (opcion) atributi

- ▶ **alt** – omogućava da se prikaže tekst u pretraživačima koji podržavaju samo tekst
- ▶ **name** – daje apletu neko simboličko ime (koristi se kod komunikacije između apleta)
- ▶ **align**, **hspace**, **vspace** – ovi atributi pozicioniraju aplet unutar web stranice.
- ▶ **align**, moguće vrednosti:
 - ▶ **left**, **right**, **middle**, **absmiddle**, **bottom**, **absbottom**, **baseline**, **top**, **texttop**
- ▶ **hspace**, **vspace**:
 - ▶ definiše veličinu praznog prostora oko apleta.

Implementacija apleta

- ▶ Aplet je bilo koja klasa koja nasleđuje klasu **Applet**.
- ▶ Klasa **Applet** je definisana u paketu **java.applet**, pa ovaj paket treba uvesti u fajl gde pišete implementaciju svog apleta.
- ▶ Sve klase u apletu moraju biti javne.

```
import java.applet.*;
```

```
public class MojApplet extends Applet  
{  
    //telo apleta  
}
```

Životni ciklus apleta

- Svaki aplet proživi 5 faza. Svaku fazu opisuje jedan metod. Svi ovi metodi su definisani u klasi **Applet**.
- **Faza inicijalizacije**
 - Tada se kreira objekat aplet i učitava se u web stranicu. Metod koji se tada izvršava je metod **init()**.
- **Početna faza**
 - To je faza kada sistem počne da izvršava aplet (odmah po inicijalizaciji ili po restartovanju apleta, tj. kad stranica sa apletom ponovo dobije fokus). Metod odgovoran za ovo je **start()**.
- **Faza iscrtavanja**
 - Nakon početne faze, ili pri svakoj promeni grafičkog u izgledu apleta. Metod odgovoran za ovu fazu je metod **paint()**.
- **Faza zaustavljanja**
 - Faza kada aplet postoji, ali više nije vidljiv (npr. korisnik gleda neku drugu web stranicu). Metod odgovoran za ovu fazu je **stop()**.
- **Faza uništavanja**
 - Faza kada sistem izbacuje aplet iz memorije. Metod: **destroy()**.

Primer apleta

```
import java.applet.*;
import java.awt.*;
public class MojApplet extends Applet
{
    public void init() { kôd za inicijalizaciju }
    public void start() { kôd za startovanje }
    public void paint(Graphics g){ kôd za crtanje }
    public void stop(){ kôd za zaustavljanje }
    public void destroy(){ kôd za uništavanje }
}
```

ActiveX kontrole

- **ActiveX** je **Microsoft-ova** tehnologija za izradu malih komponenti (kontrola) u okviru Web strane – nije programski jezik, već mnoštvo integrisanih objekata napisanih na različitim jezicima u Windowsovom okruženju.
- **ActiveX** objekti (kontrole) omogućavaju dodavanje programa Web stranici za rešavanje mnoštva zadataka: od prikazivanja poruka do npr. mogućnosti prepoznavanja govora u Web stranicama.

Active X kontrole

- ActiveX kontrole mogu biti izvršavane samo na *Windows* operativnim sistemima.
- ActiveX objekti nisu bezbedni, jer moraju da se nalaze na korisnikovom disku da bi ga bilo moguće koristiti - imaju pristup svim resursima sistema
- Zlonamerni ActiveX objekat može uneti virus i oštetiti ili uništiti podatke na disku korisnika

ActiveX kontrole

- Ako korisnik poseti Web stranicu koja koristi ActiveX objekat, a nema instaliran objekat na svom disku, Web server će pokušati da ga pošalje na korisnikov računar
 - Zavisno od korisnikovih bezbednosnih podešavanja, čitač Weba će prikazati okvir za dijalog, koji upozorava korisnika o preuzimanju ActiveX objekta
- Preporuka: koristiti samo svoje objekte, one koje su napravili poznati i pouzdani programeri i one koji su preuzeti sa pouzdanih Web lokacija (kao što je Microsoft)

JavaScript

- Programski jezik inicijalno nastao za programiranje klijentske-strane web aplikacije
- Promovisao ga je NetScape 1995. godine
- Standardizovan kao ECMAScript od strane ECMA (European Computer Manufacturers Association)
- Osobine JavaScript-a
 - Interpretatorski jezik
 - Objektno-orijentisani jezik
 - Podržava neke elemente funkcionalnog programiranja
 - Dinamički i slabo tipiziran jezik

Skripte u HTML-u

- Kratki parčići koda koji se ugrađuju u sam HTML
- Umetanje JavaScript koda u HTML:

```
<script type="text/javascript" language="javascript">  
    //javascript kod  
    document.write("Hello World!")  
</script>
```

- Ukoliko je kod koji treba izvršiti glomazniji, pišu se funkcije u posebnom .js fajlu, pa se u umetnutom kodu samo pozivaju.
- U tom slučaju veza sa spoljašnjim .js fajlom se ostvaruje navođenjem u head delu skripte:

```
<script src="filename" type="text/javascript">  
</script>
```

Tipovi podataka i promenljive u JavaScriptu

- Iako je slabo tipiziran jezik, postoje sledeći tipovi podataka:

- Number, Boolean, String, Array, Object, Function, Null, Undefined

- Definicija promenljivih:

```
var num = 5;
```

```
var str = "Ovo je string";
```

```
var x;
```

```
let y = 6;           //važi samo u bloku u kojem je def.
```

Polja u JavaScriptu

- Polja imaju promenljivu veličinu

- Definicija polja:

```
var a = []; //prazno polje  
var b = [1,2,3];
```

- Dodavanje elementa na kraj polja:

```
a.push(5);
```

- Dodavanje elementa na početak polja:

```
a.unshift(5);
```

- Brisanje elementa s kraja polja:

```
a.pop();
```

- Brisanje elementa sa početka polja:

```
a.shift();
```

Programske strukture u JavaScriptu

- Postoje sve programske strukture kao u programskom jeziku C.

Funkcije u JavaScriptu

- Definicija funkcije:

```
function <name>(<parameters>)  
{  
    //body  
}
```

- Primer:

```
function sum(a,b)  
{  
    return a+b;  
}
```

Klase u JavaScriptu (ES5 i niže)

- Definicija konstruktora klase:

```
function <ime>(<parameters>)  
{  
    //telo konstruktora - za pristup  
    //atributima koristi referencu this  
}
```

- Primer konstruktora klase:

```
function Person(name)  
{  
    this.name = name;  
}
```

- Kreiranje objekta klase:

```
let p = new Person("Petar");
```

Klase u JavaScriptu (ES5 i niže)

- Za dodavanje novih svojstava i metoda klasi koristi se skriveni objekat **prototype**.
- Dodavanje metoda klasi:

```
<ime_klase>.prototype.<ime> = function(<parameters>)  
{  
    //body  
}
```

- Primer dodavanja metode:

```
Person.prototype.show = function()  
{  
    alert(this.name);  
}
```

- Poziv metode klase:

```
p.show();
```


Klase u JavaScriptu (ES6)

- Definicija klase:

```
class <ime>
{
    // class methods
    constructor() { ... }
    <ime_mettode> { ... }
    ...
}
```

- Primer klase:

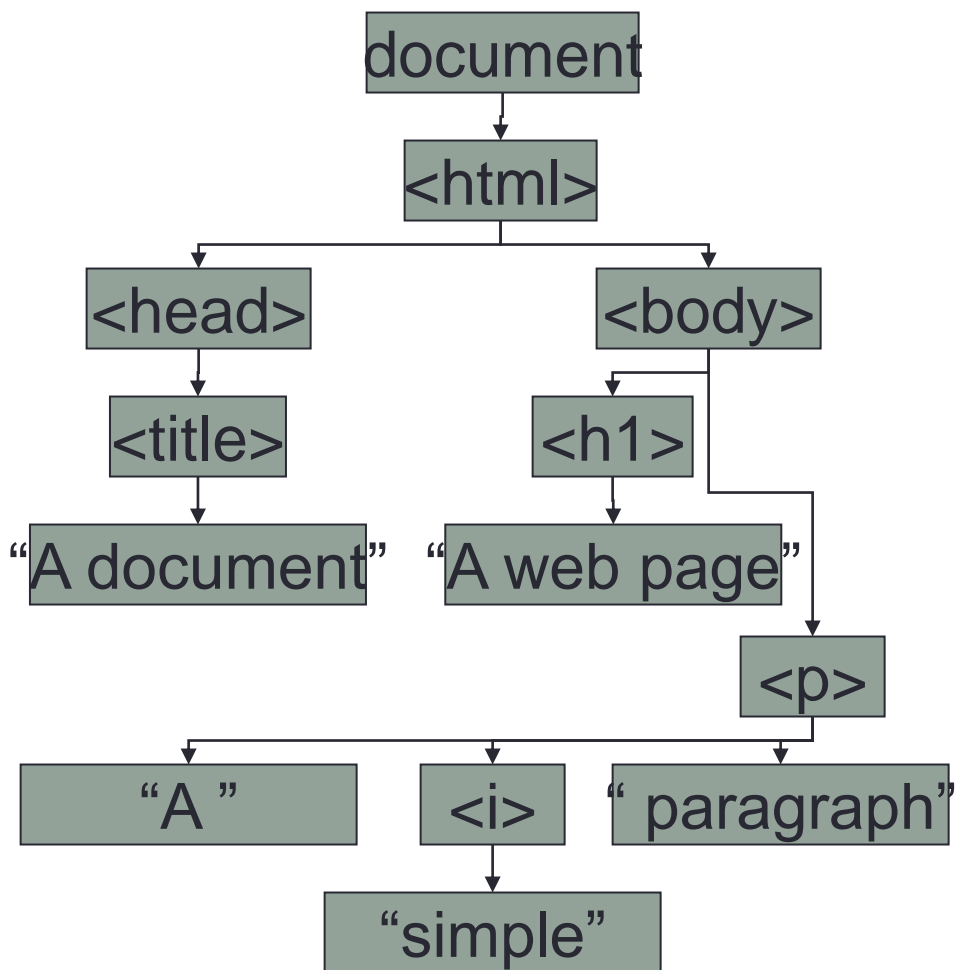
```
class Person
{
    constructor(name) { this.name = name; }
    show { alert(this.name); }
}
```

Manipulacija sadržajem HTML dokumenta pomoću DOM-a

■ HTML kod:

```
<html>
<head>
  <title>A Document</title>
</head>
<body>
  <h1>A web page</h1>
  <p>A <i>simple</i>
    paragraph</p>
</body>
</html>
```

■ DOM:



Manipulacija sadržajem HTML dokumenta pomoću DOM-a

```
function addParagraph()  
{  
    var paragraph = document.createElement("p");  
    var text = document.createTextNode("Novi pasus");  
    paragraph.appendChild(text);  
    document.body.appendChild(paragraph);  
}
```

Implementacija event-handlera

- Definicija event-handlera:

```
<element event='//some JavaScript'>
```

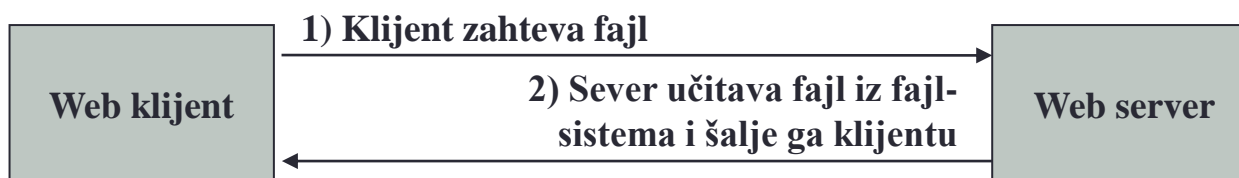
- Primer:

```
<button onClick=  
    'alert("dugme pritisnuto");'>Dugme 1</button>  
<button onClick=  
    'addParagraph();'>Dugme 2</button>
```

PROGRAMIRANJE SERVERSKE STRANE WEB APLIKACIJA

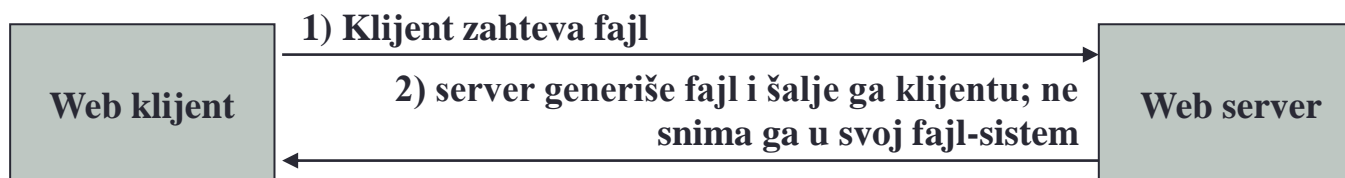
Statičke i dinamičke web strane

- Statičke – uskladištene na strani web servera i isporučuju se na zahtev korisnika



- Isporuka statičkih sadržaja -

- Dinamičke – generišu se na strani servera i odmah šalju korisniku (ne skladište se na strani servera)



- Isporuka dinamičkih sadržaja -

CGI programi

- Najstarija tehnologija za serversko programiranje:
Common Gateway Interface (CGI) programi
- CGI omogućava Web serverima da pokreću eksterne programe koji će generisati odgovor na HTTP zahteve
- CGI programi mogu biti implementirani u proizvoljnom programskom jeziku. Oni
 - Čitaju podatke sa svog standardnog ulaza,
 - Upisuju rezultat na svoj standardni izlaz

CGI programi

- Po pristizanju zahteva, web server pokreće CGI program i na njegov standardni ulaz upisuje ulazne podatke preuzete iz zahteva.
- Po završetku rada programa, preuzima rezultate CGI programa sa njegovog standardnog izlaza (što može biti generisani HTML) i šalje klijentu.
- Kao jezik implementacije CGI programa najčešće je korišćen Perl (krajem prethodnog stoleća)

Java servleti

- Po načinu rada vrlo slični CGI programima – nezavisni programi koji se izvršavaju na serveru i generišu rezultujuće stranice
- Za implementaciju servleta koristi se isključivo programski jezik Java

Kreiranje HTML stranica korišćenjem serverskih skript jezika

- Na serveru su uskladišteni dokumenti koji predstavljaju mešavinu html koda i skripti koje služe za generisanje stvarnih html dokumenata.
- Primer generisanja strane na osnovu PHP fajla:

```
<html>
<head>
  <title> PHP </title>
</head>
<body>
  <?php print(
    "Hello World!");
  ?>
</body>
</html>
```



```
<html>
  <head>
    <title> PHP </title>
  </head>
  <body>
    Hello World
  </body>
</html>
```

Ugradnja PHP koda u HTML

- U okviru HTML koda:

```
<script type="php">  
    ...PHP kod ...  
</script>
```

- Ili u eksternom fajlu:

```
<script type="php" src="phpkod.php">  
</script>
```

- Češće se koristi uprošćena sintaksa

```
<?php ...PHP kod... ?>
```

- odnosno

```
<?php include("phpkod.php") ?>
```

Tipovi podataka i promenljive u PHPu

- Kao i JavaScript, dinamički i slabo tipiziran jezik
- Tipovi podataka:
 - Skalarni: boolean, integer, float, string
 - Složeni: array, object
 - Specijalni: resource (pokazivač na fajl, na vezu sa bazom podataka i sl.), Null
- Promenljive:
 - Imena promenljivih počinju simbolom \$ iza kojeg sledi slovo ili _
 - Promenljive se ne deklarišu

Polja u programskom jeziku PHP

- Indeksirana polja:

```
$niz[0] = 1;
```

```
$niz[1] = 5;
```

```
$niz[] = 10; // $niz[2]=10
```

- Asocijativna polja – elementima se pristupa po ključu:

```
$ocena["OOP"] = 10;
```

```
$ocena["Matematika"] = 8;
```

```
...
```

Izrazi i naredbe u PHPu

- Izrazi i upravljačke strukture imaju isti format kao u programskom jeziku C.

Funkcije u programskom jeziku PHP

- Definicija funkcije:

```
function <ime>(<parametri>)  
{  
    //telo  
}
```

- Primer:

```
function sum($a, $b)  
{  
    return $a + $b;  
}
```

Klase u PHP-u

- Definicija klase:

```
class <ime>
{
    // class attributes
    [<pravo_pristupa>] var <ime_atributa>
    // class methods
    [<pravo_pristupa>] function __construct() { ... }
    function <ime_mettode> { ... }
    ...
}
```


Klase u PHP-u

- Primer klase:

```
class Person
{
    var $name;
    function __construct($name)
    {
        $this.$name = name;
    }
    function show()
    {
        print($this.$name);
    }
}
```

Objekti u PHP-u

- Kreiranje i korišćenje objekata:

```
$p = new Person("Petar");  
$p->show();
```

Komunikacija između klijenta i servera

- Kada klijent traži od servera da mu generiše i pošalje neku stranu poziva **get** metodu,
- Kada klijent šalje podatke serveru poziva **post** metodu
- Parametri ovih metoda se prenose preko superglobalnih promenljivih **\$_GET** i **\$_POST**
- **\$_GET** i **\$_POST** su asocijativni nizovi.

Komunikacija između klijenta i servera - primer

```
<form action="Primer.php" method="get" >
    Korisničko ime:
    <input type="text" name="ime"/>
    ...
    <input type="submit" name="submit" value="prosledi"/>
</form>
```

- **Primer.php:**

```
<html>
<head>
    <title> Primer </title>
</head>
<body>
    <?php print( $_GET["ime"] ); ?>
</body>
</html>
```

WEB SERVIZI

Web servisi

- Aplikacija smeštena na nekom web serveru koja može da se pozove sa različitih klijenata
 - Klijent u XML formatu šalje zahtev koji sadrži ime procedure koju želi da pozove i parametre
 - Server u XML formatu vraća rezultate
- Web Servisi mogu da se javno objave na jedinstvenoj lokaciji gde se nude kao usluge.
- UDDI (**U**niversal **D**escription, **D**iscovery and **I**ntegration) je centralizovana lokaciju koja obezbeđuje mehanizam za registrovanje i pronalaženje Web servisa.
- Web servisi mogu biti i lokalni
 - Koriste se često u Web aplikacijama kod kojih je cela logika smeštena na klijentu, a preko web servisa se samo komunicira sa bazom podataka koja je smeštena na serveru

Jezici koji se koriste za rad sa Web servisima

- WSDL (**W**eb **S**ervice **D**escription **L**anguage) – jezik zasnovan na XML-u, služi da opiše kako se koristi web servis
- SOAP (**S**implified **O**bject **A**ccess **P**rotocol)– protokol za razmenu poruka – definiše format zahteva (request), i format odgovora (response)