

# HTML, CSS i JavaScript

G04Code

Marko Cvijanović  
Mihailo Bubnjević

# HTML

- HyperText Markup Language
- Definiše strukturu sadržaja na nekoj veb stranici
- “Hypertext” se odnosi na linkove koji povezuju veb stranice
- HTML koristi “markup” da anotira tekst, slike i druge vrste sadržaja kako bi se prikazali u browseru na odredjen način

# Markup

- HTML tagovi su u glavnom par otvarajućeg **<p>** i zatvarajućeg **</p>** taga gde se između njih dodaje njihov sadržaj
- Sadržaj može biti tekst ili drugi tagovi
- Postoje i slučajevi "self-closing" tagova kao što su **<img>** , **<br>**, **<meta>** , **<input>** koji ne moraju da imaju closing tag
- Svaki html dokument počinje i završava se sa **<html>** tagom koji sadrži
  - **<head>** - U kojem definišemo metapodatke o HTML dokumentu kao što su naslov i ikonica u tabu, stilove, skripte i ostali metapodaci koji se neće prikazivati
  - **<body>** - koji definiše telo HTML dokumenta i sadrži sav tekst, slike, naslove i ostali sadržaj

# Markup

- Tagovi takodje mogu da sadrže attribute unutar svog otvarajućeg taga

```

```

- Atributi su "key value" parovi koji definišu izgled ili ponašanje nekog taga
- Atributima možemo da definišemo informacije vezane za taj tag kao što su identifikatori za stilizovanje (id, class), linkove (href), tip (type) i druge

# Markup

```
<!DOCTYPE html>
<html>
  <head>
    <title>This is a title</title>
  </head>
  <body>
    <div>
      <p>Hello world!</p>
    </div>
  </body>
</html>
```

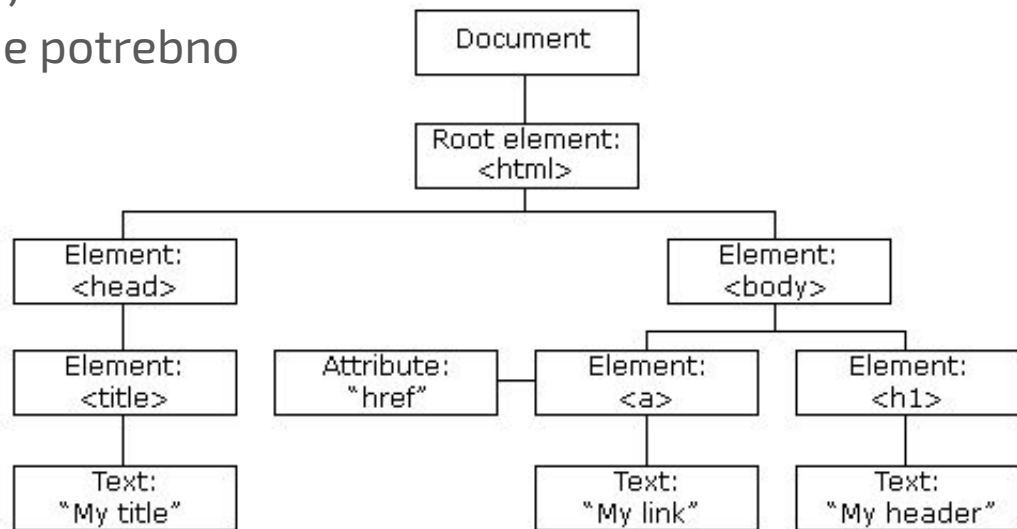
# HTML5

- U prošloj verziji HTML-a nije postojao način da se integriše video unutar
- Rešenje za to su bili eksterni plugini, među najpoznatijima Flash Player
- Korisnik sajta koji sadrži eksterne plugine morao je da ih skine
- HTML5 je 2008. rešio ovo dodavajući nove tagove da plugini više nisu bili potrebni
- Neki od novih tagova su:
  - <audio>
  - <video>
  - <canvas>
  - <article>
  - <nav> ...



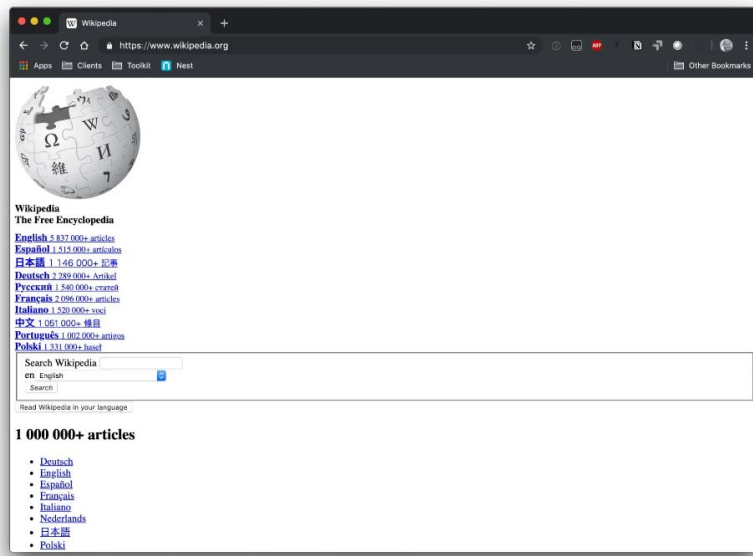
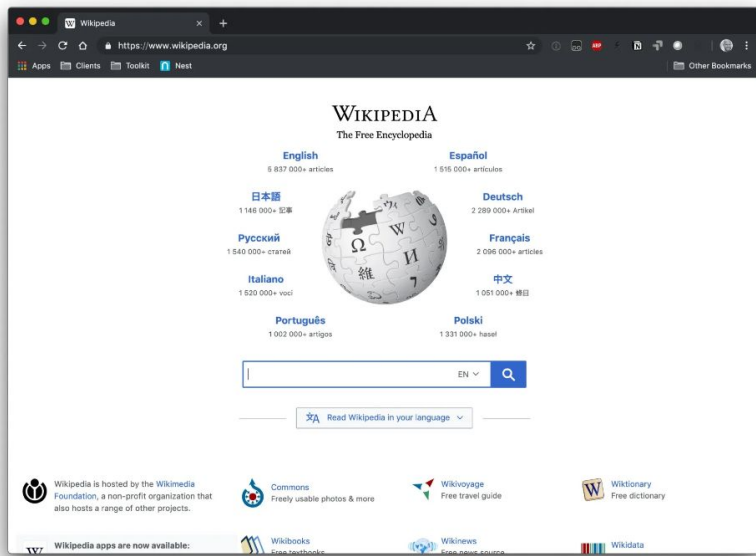
# HTML DOM

- Kada se veb stranica učitava, browser pravi **Document Object Model** za tu stranicu
- HTML Dom definiše standard za pristupanje i manipulaciju dokumentima
- DOM je konstruisan kao drvo objekata
- Sa ovim JavaScript ima sve što je potrebno da se pravi dinamički HTML



# CSS


- **Cascading Style Sheets**
- Sam HTML browser će prikazati kao crno beli dokument
- Za prezentaciju tog dokumenta koristimo CSS, gde možemo da utičemo na raspored, izgled, boju i poziciju HTML elemenata





# CSS

- CSS se sastoji od određenih pravila (css rules)
- Tipično pravilo se sastoji od
  - **Selektora** - definiše koji element ili grupu elemenata iz HTML dokumenta želimo da menjamo
  - **{ }** - za definisanje deklarativnog bloka koji sadrži svojstva za definisanje izgleda
  - **Svojstva (Properties)** - **{ key : value; }** key value parovi koji definišu izgled ili ponašanje

```
h1 { color:  red; }
```

# CSS

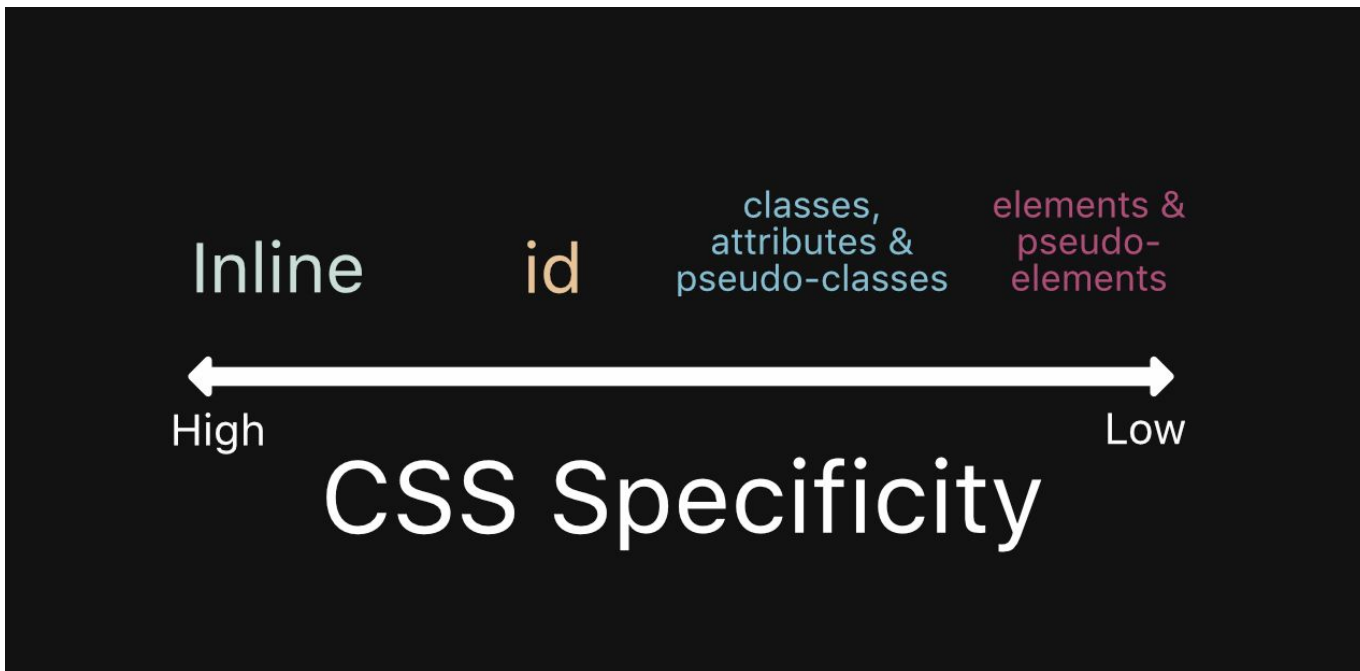
- Postoje 3 vrste selektora
  - **Po tipu** - odnosi se na sve HTML elemente tog tipa
  - **Po klasi** - odnosi se na sve HTML elemente koji imaju definisanu tu klasu
  - **Po id-u** - odnosi se na jedan HTML element sa tim id-em

# CSS

- **Stylesheet Cascade** - šta će se desiti ako imamo više definisanih stilova nad istim elementom
- Kada se to desi kreira se hijerarhija pravila
- Ona pravila koja su specifičnija imaju prioritet
- Redosled pravila je bitan
- Neki elementi mogu da nasleđuju stilove od roditeljskih elemenata

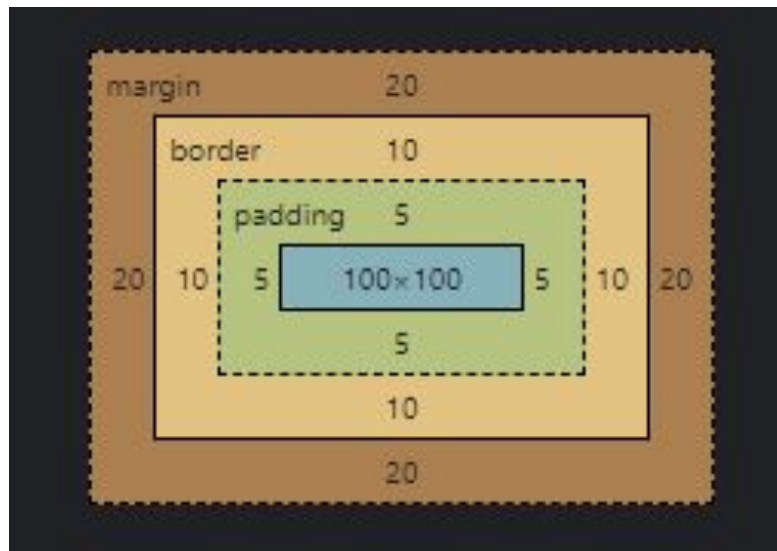
# CSS

- Princip specifičnosti



# CSS

- Svaki HTML element se ponaša kao pravougaonik
- Svaki element ima oko sebe
  - Padding
  - Border
  - Margin

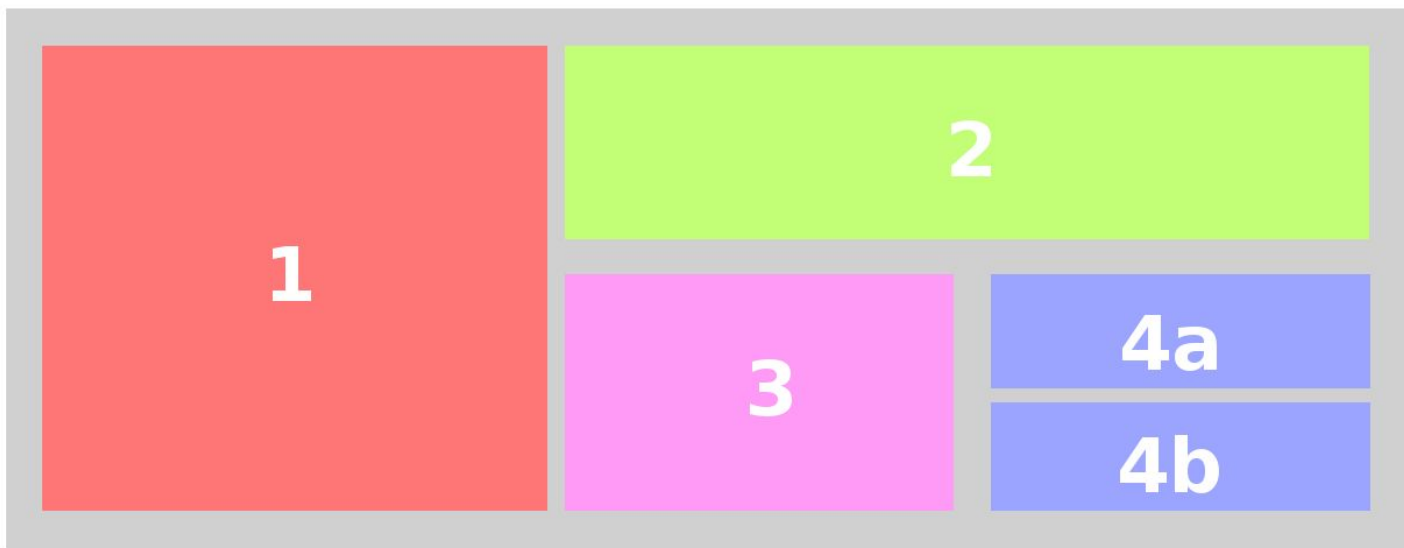


# CSS

- Veličina ovih “okvira” se može definisati
  - Sa Apsolutnim dužinama
    - px - pikseli
    - mm - milimetri
    - cm - centimetri
    - in - inči
  - Sa Relativnim dužinama
    - vh/vw - relativno na 1% visine(h) ili dužine(w) prozora
    - % - postotak relativan na roditeljski element
    - em - relativan na veličinu fonta tog elementa

# CSS

- Da ne bismo svaki element eksplicitno pozicionirali koristimo **layout**-e
- **Flexbox layout** koristimo da ređamo elemente horizontalno ili vertikalno
- Flex container širi elemente tako da najbolje popuni dostupan prostor



# CSS

- **Grid layout** - ređa elemente u redove i kolone





# CSS

- Napredne teme css-a:
  - Pseudo selektori
  - Logika unutar css-a
  - variable
  - media query
  - animacije ...

# JavaScript

- Programski jezik za kreiranje interaktivnih web stranica zajedno sa HTML i CSS-om
  - HTML - definiše sadržaj stranice
  - CSS - definiše raspored i izgled stranice
  - JS - definiše ponašanje stranice
- Interpretiran tj. JIT(just-in-time compiled)
- Single-threaded
- First-class functions

# JavaScript



## ATWOOD'S LAW

ANY APPLICATION THAT CAN  
BE WRITTEN IN JAVASCRIPT  
WILL EVENTUALLY BE  
WRITTEN IN JAVASCRIPT

-JEFF ATWOOD

# JavaScript - Tipovi

- Primitivni:
  - number
  - bigint
  - string
  - boolean
  - null
  - undefined
  - symbol
- Sve ostale vrednosti su objekti
  - Svi primitivni tipovi osim null i undefined imaju wrapper objekte tipove (Number, String, Boolean..)

# Konverzije primitivnih tipova

Vrednost	String	Number	Boolean
undefined	"undefined"	NaN	false
null	"null"	0	false
true	"true"	1	
false	"false"	0	
""		0	false
"1.2"		1.2	true
"one"		NaN	true
0	"0"		false
NaN	"NaN"		false
Infinity	"Infinity"		true
5	"5"		true

# Konverzije primitivnih tipova

- U izrazima za koje se očekuje da vrate *boolean* uvek će se izvršiti konverzija

```
if("asd"){  
    console.log("!");  
}
```

- Možemo raditi i eksplicitnu konverziju

```
Number("23");
```

# JavaScript

- Falsy vrednosti

- false
- 0
- ""
- null
- undefined
- NaN

- Truthy vrednosti

- true
- bilo koji objekat
- bilo koji string koji je duzi od 0
- bilo koji broj različit od 0

# JavaScript

- Šta će ispisati sledeći izraz

```
console.log(3 + 2 + "7");
```



# JavaScript

- Definisanje varijabli
  - **var** - varijable definisane sa var su dostupne kroz celu funkciju u kojoj su definisani
  - **let** - samo dostupne unutar bloka u kom su definisane
  - **const** - za definisanje konstanti

```
function varExample() {  
  var x = 1;  
  
  if (true) {  
    var x = 2;  
    console.log(x); // will print 2  
  }  
  console.log(x); // will print 2  
}
```

```
function letExample() {  
  let x = 1;  
  
  if (true) {  
    let x = 2;  
    console.log(x); // will print 2  
  }  
  console.log(x); // will print 1  
}
```

# JavaScript

- Funkcije u JS su first-class objekti:
  - Mogu da se dodeljuju u varijable, objekte i nizove
  - mogu da se prosleđuju kao parametri
  - mogu da budu povratna vrednost neke druge funkcije

```
let fn = function doSomething() {}           // u varijabli

let obj = { doSomething : function doSomething(){} } // u objektu

arr.push(function doSomething() {})          // u nizu
```

# TypeScript

- Tipiziran
- Ponaša se kao kompajler za JavaScript
- Čitljiviji i lakše se održava naspram JavaScript
- Podržava apstrakciju putem interfejsa
- Podržava anotiranje putem dekoratora
- Podržava Generic tipove