

# 1. WWW - osnove (HTTP, HTML, URL)

HTTP je skraćenica za HyperText Transfer Protocol. Ovo je najpoznatiji protokol koji se danas koristi. Ovaj komunikacioni standard koristi web pretraživač da bi komunicirao sa serverom web sajta koji pretražujemo (ili koji smo otvorili).

WWW je skraćenica za World Wide Web i predstavlja sistem međusobno povezanih HyperText dokumenata na internetu kojima se pristupa preko Web browsera (čitača).

Internet predstavlja "mrežu svih mreža" tj. svetsku javno dostupnu mrežu povezanih računarskih mreža koje prenose podatke korišćenjem IP standarda.

HTML definiše strukturu i sadržaj strane (web strana je dokument pisani u HTML-u).  
Web aplikacija je aplikacija kojoj se pristupa preko web čitača korišćenjem Interneta.  
Web sajt je skup web strana u okviru nekog domena.

HTTP i WWW su sastavni deo svih URL-a web sajtova.

HTTP se koristi za transfer podataka sa/ka web sajtu. (komunikacija web servera i web browsera)

WWW ukazuje na to da se radi o web sajtu i da se koristi HTTP protokol

http://nesto.com

www.nesto.com

http://www.nesto.com

} svi vode  
do istog  
sajta

URL je skraćenica od Universal Resource Locator i predstavlja kompletnu web adresu koja se koristi za lociranje određene web strane. Domen predstavlja ime sajta, a URL nas vodi do neke web strane u okviru tog sajta. U okviru URL-a se nalazi ime domena, protokol koji se koristi (HTTP, HTTPS, ...), prefiks www i ostale komponente potrebne za lociranje web strane.

Internet protokoli

## 2. Web aplikacije

Web aplikacija je aplikacija kojoj se pristupa preko web čitača korišćenjem interneta ili INTRANET mreže.

Razlikuju se od ostalih aplikacija po tome što nije potrebno instalirati ih, već im se pristupa preko web čitača.

Primeri web aplikacija: Facebook (društvena mreža)  
Wikipedia  
Flickr (deljenje slika)  
Mibbit (četovanje)

Web aplikacije se baziraju na klijent-server modelu, gde se klijent (ili korisnički interfejs) izvršava u web čitaču. Nije potrebno bilo kakvo instaliranje softvera, na klijentskoj strani, da bi se videle promene na web aplikacijama.

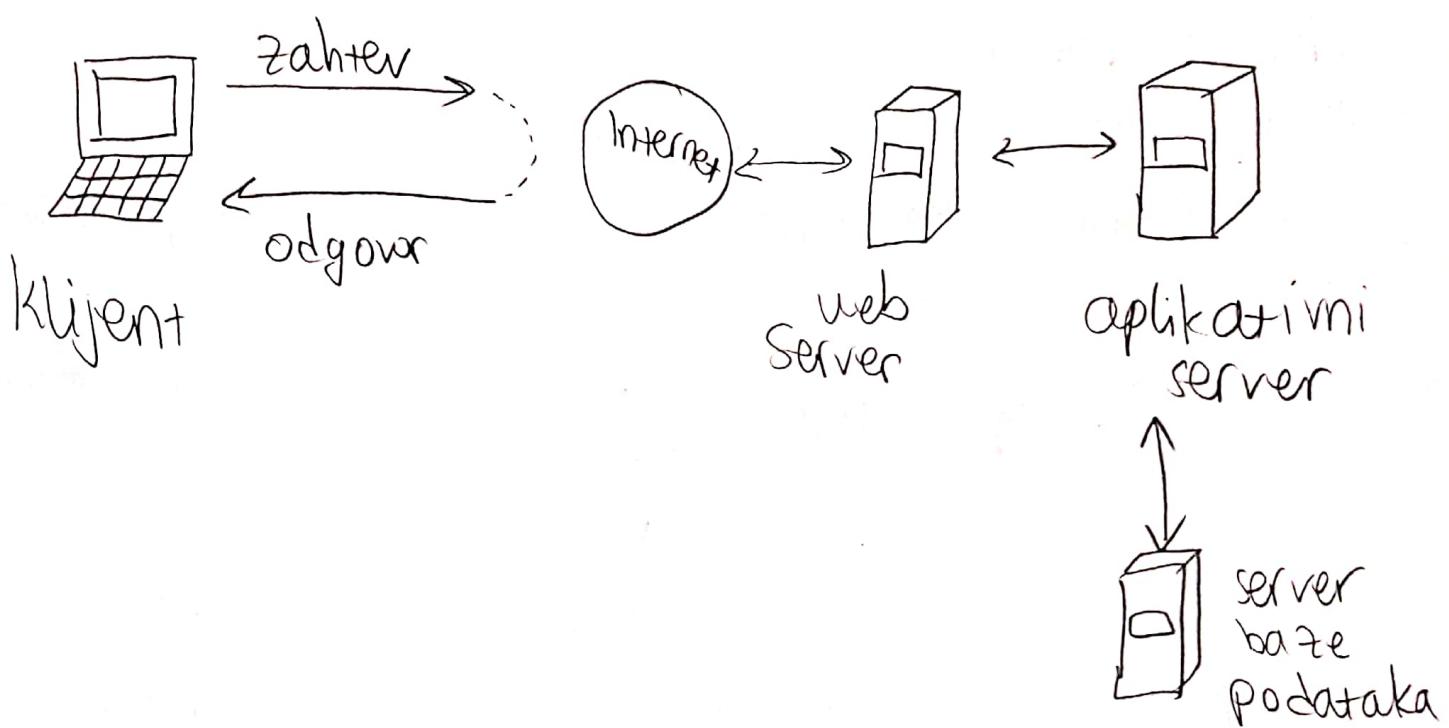
Nije moguće precizno definisati razliku između web aplikacije i web sajta.  
Web aplikacije se najčešće organizuju u rangove (tier) i obično ih ima 3:

1. Prezentacija → Web čitač

2. Aplikacija → sendžin koji koristi neku Web tehnologiju (ASPGI, PHP, ) Node.js

3. Skladište → baza podataka

Web čitač šalje zahteve srednjem rangu, koji ih servira tako što šalje zahteve i naredbe bazi podataka i generiše korisnički interfejs.

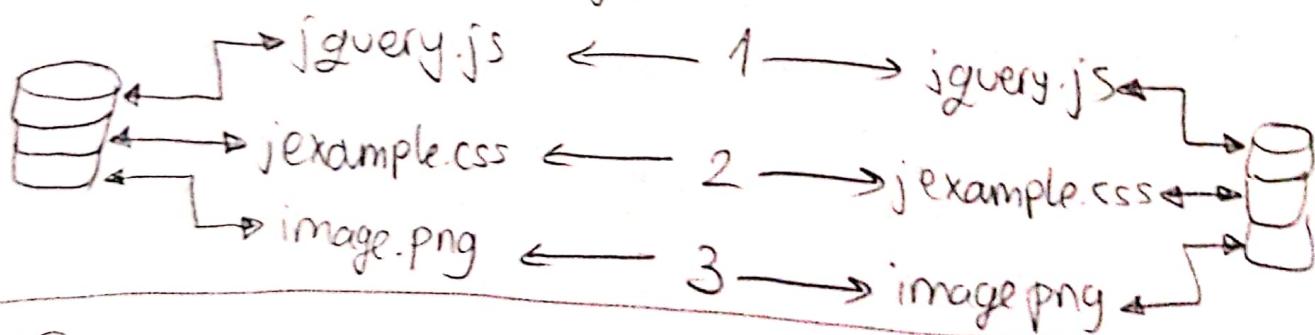


### 3. Protokoli (HTTP 1.0/1.1/2.0 i WebSocket)

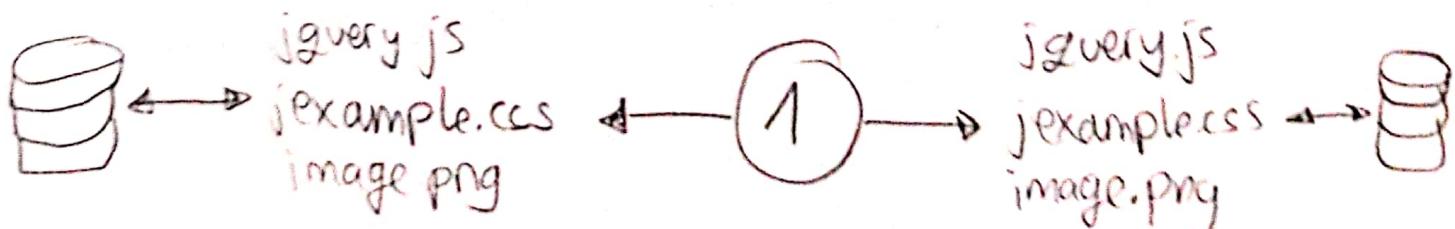
#### HTTP 2.0

- binarni protokol (ne tekstualni)
- u potpunosti multipleksiran (HTTP 2.0 može poslati više zahteva za podacima paralelno preko 1 TCP veze. Ovo je najnaprednija karakteristika HTTP 2.0 protokola jer omogućava preuzimanje podataka (web files) sa jednog servera preko ASync režima.)

#### HTTP 1.1 TCP konstrukcija



#### HTTP 2.0 TCP konekcija



→ vrši kompresiju zaglavlja (vrši kompresiju velikog broja redundantnih okira zaglavlja. Koristi **HPACK** specifikaciju za jednostavan i siguran pristup pri kompresiji zaglavlja. HPACK vrši kompresiju svakog pojedinčnog zaglavlja pre prenosa na server, koji zatim traži kodiranu informaciju u listi prethodno prenesenih vrednosti zaglavlja da bi rekonstruisao potrebne informacije zaglavlja.)

Većina modernih web čitača podržava HTTP2.0.

- HTTP 1.0 → pravi se nova veza za svaki zahtev → PROBLEM
- Podaci o podacima ↑
- Protokol koji podržavaju web čitači
  - U zaglavlju se nalaze polja sa metapodacima o zahtevu i odgovoru (broj verzije HTTP-a, statusni kod, tip sadržaja)
  - Odgovor nije ograničen na hypertext (Content Type u zaglavlju pruža mogućnost za prenos i drugih tipova datoteka osim običnog HTML-a npr. skripte, slike, ...)
  - Podržane metode: GET, POST, HEAD
  - Veza se prekida odmah nakon odgovora

### 3. Protokoli (HTTP 1.1 i WebSocket)

#### HTTP 1.1

- Uvedene optimizacije performansi i poboljšanje karakteristika (blokirani transferi, uporne i protične veze → persistent and pipelined conn., kompresija/dekompresija, pregovori sadržaja, virtualni hosting (server sa 1 IP adresom (hosting) je host za više IP domena), brži odziv, i ušteda propusnog opsega do davanjem kesa)
- Podržane metode: GET, POST, HEAD, PUT, DELETE, TRACE, OPTIONS
- Veze su dugotrajne (priroda veza: dugotrajna)

#### WebSocket → kompatibilan HTTP serveru

- protokol koji omogućava full-duplex komunikaciju u jednoj TCP konekciji
- protokol 7. nivoa OSI modela (kao i HTTP ali se razlikuje od HTTP-a) i zavisi od protokola 4. nivoa (TCP)
- dizajniran da radi preko HTTP portova 80 i 443 (za SSL) kao i da podrži HTTP proksije (proxies) i posrednike.

- Websocket handshake koristi HTTP Upgrade zaglavje (deo u zaglavju) za promenu protokola iz HTTP u WebSocket.
- Omogućava interakciju između web čitača (ili druge klijentske aplikacije) i web servera sa manjim "troškovima" što olakšava prenos podataka sa i na server.

## 9. Progresivno poboljšanje

To je strategija u web dizajnu koja koristi web tehnologije na složenit način, odnosno omogućuje svakom da pristupi osnovnom sadržaju i funkcionalnosti web-stranice, kroz bilo koji web čitač ili sa bilo kakvom internet konekcijom, ali omogućava i pristup naprednoj verziji stranice za korisnike sa naprednjim web čitačima ili sa većim protokolom podataka.

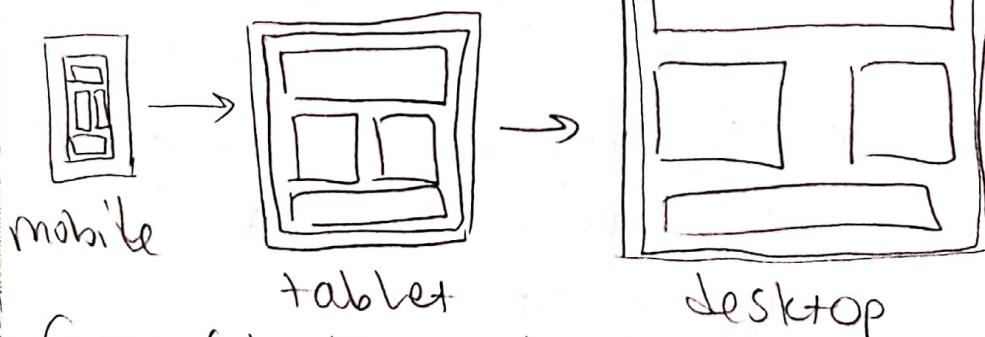
Tehnika se oslanja na ranije - popularnu strategiju "elegantna degradacija" (Graceful Degradation), gde dizajneri kreiraju web stranice namenjene najnovijim web pretraživačima, ali imaju u vidu i njihove ranije verzije. Elegantna degradacija treba da dopusti stranici da "degradira", odnosno da održi svoju prezentabilnost i pored toga što neke tehnologije koje dizajner želi da koristi nisu implementirane na korisnikovoj verziji web čitača.

PE čine sledeći osnovni principi:

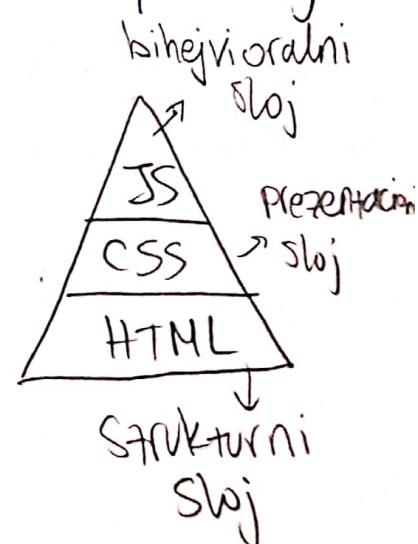
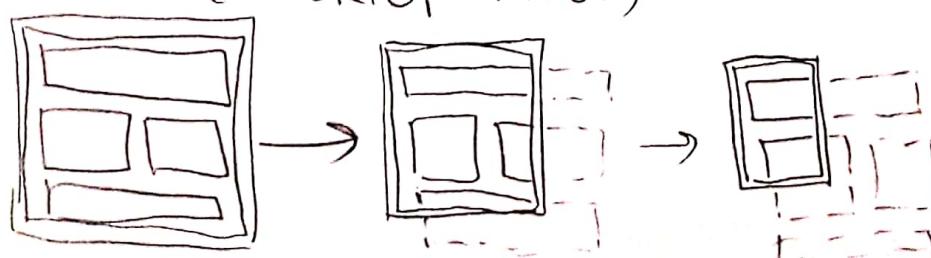
1. Osnovni sadržaj treba da bude dostupan na svim pretraživačima
2. Osnovna funkcionalnost treba da bude dostupna na svim browserima
3. Sadržaj je označen bez suvišnih tagova i na semantički način
4. Napredni izgled stranice se oslanja na eksterno povezani CSS
5. Napredna funkcionalnost je ugrađena koristeći nenumetljiv (unobtrusive) JS
6. Podrazumjedna podešavanja na web čitačima krajnjih korisnika se poštuju

Progressive Enhancement

(Mobile First)



Graceful Degradation  
(Desktop First)



## III. Responsive dizajn

Responsive dizajn se bavi korišćenjem HTML-a i CSS-a za automatsku promenu veličine strane, smanjivanje, sakrivanje, uvećavanje web strane kako bi dobro izgledala na svim vrstama uređaja (tabletima, telefonima, desktop računarima).

Responsive web dizajn je pristup web dizajnu koji čini da se web stranice dobro prikazuju (renderuju) na različitim veličinama ekrana i na različitim uređajima.

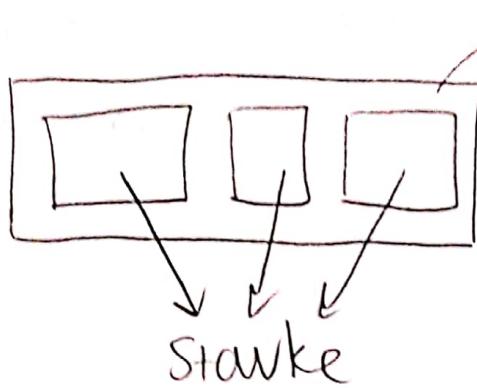
## 10. Flexbox dizajn

Ovaj dizajn ima za cilj da obezbedi efikasniji način za postavljanje, poravnanje i distribuciju prostora među stanicama u kontejneru, čak i kada je veličina stanki nepoznata ili dinamična.

Glavna ideja flex dizajna je da se kontejneru omogući da menja širinu/visinu i redosled svojih stanki (stanki se nalaze unutar kontejnera) kako bi najbolje popunio raspoloživi prostor (da se prilagodi svim vrstama uređaja i veličinama ekranova). Flex kontejner proširuje ili smanjuje

stavke kako bi popunio slobodan prostor ili  
sprečio prelivanje (overflow).

Flexbox je pogodan za male raspored (layout-ove),  
a za veće layout-ove se koristi Grid layout.



• item {

order: 2

}

kontejner

• kontejner {  
display: flex,  
}

• kontejner {  
flex-direction: row  
column  
row-reverse  
column-reverse  
}  
y Flex-grow: 1

## 12. JS opšte

JS je univerzalni jezik klijent strane. Script se izvršava na klijentskoj mašini pošto se dokument učita ili u nekom drugom trenutku (npr. kada se aktivira link). Skripte pružaju autoru sredstva za proširenje funkcionalnosti i interaktivnosti HTML dokumenta:

- dinamički menjati sadržaj dokumenta
- validacija korisničkih podataka u obrazcu, kako bi se detektovale greske pre slanja podataka serveru
- komunikacija sa serverom (AJAX, Websockets)
- obrada događaja koji se odnose na dokument kao što su učitavanje i zatvaranje dokumenta, pokreti kursora miša, fokus elementa, itd.

JS je najpopularniji jezik Web-a, zastavljen na klijentu (Web čitaci), na serveru (Node.js), ali i van Web-a: PDF, desktop widgeti. Standardizovan je ECMAScript specifikacijom.

JS je prototipski skript jezik dinamičkih tipova.

→ Prototipski bazirano programiranje → stil OO programiranja gde se nasleđivanje postiže kloniranjem postojećih objekata, koji služe kao prototipi

→ Dinamički tipovi: ista promenljiva može da menja tip podatka

Primer

```
var name = "Hello";  
name = 4;
```

JS objekti su asocijativni nizovi

Obj-X  $\Leftrightarrow$  Obj['x']

JS je CASE-SENSITIVE!

Skripta može biti interna (u okviru HTML-a) ili eksterna (u zasebnoj datoteci). DEKLARISANJE Skripte se vrši pomoću script elementa.

<body>  $\rightarrow$  Interne

```
<script type="text/javascript">  
    alert("Bla");  
</script>
```

<body>

<body>  
<script src="file.js" type="module">  
</script>  
</body> /

Ekserne

# 13. JS funkcije

Funkcije su jedan od najbitnijih konceptova u JS-u. One predstavljaju deo koda koji može više puta da se izvršava, pozivanjem preko određenog identifikatora

→ direktna deklaracija      → poziv:  
funkcija nekaFunkcija(arg1, arg2, arg3) }      nekaFunkcija  
//...  
} return povratnaVrednost;

Koristi se ključna reč function, nakon koje sledi ime funkcije sa listom argumentata unutar zagrade.

Funkcija može da se deklariše tako da ima globalni opseg, ali takođe i ugnježđeno unutar neke druge f-je, u kojem slučaju može da se koristi samo unutar te (spoljašnje) funkcije.

Funkcija može da se poziva i pre deklaracije zato što JS implicitno sve deklaracije "vidi" pre izvršnog dela koda. Osim preko direktnе deklaracije, f-je možemo definisati:

var nekaFunkcija = function (arg1, arg2, arg3) {  
 //...  
 return povratnaVrednost; → funkciski  
}      izraz

Promenljivoj neka funkcija se dodjeljuje referenca na neku anonimnu f-ju, čija deklaracija, upravo sledi. Funkcija se poziva na isti način. Razliko u tome što se ne može pozvati pre deklaracije, jer u tom trenutku promenljiva neka funkcija ne ukazuje ni na šta. Funkcija na koju pokazuje promenljiva ne mora da bude anonimna, može imati ime, ali se ono ne može koristiti van konteksta same funkcije. Unutar f-je, ime se može koristiti, i na ovaj način možemo implementirati rekurziju.

```
var faktorijel = function fakt+(n){  
    if(n==0) return 1;  
    return n*fakt+(n-1);  
}
```

JS NE podržava preklopljavanje f-ja. F-ja se može pozvati sa bilo koliko argumenta. Tada se pristupa promenljivoj arguments, koja predstavlja niz svih argumenta sa kojim je pozvana f-ja.

```
function zbir(){  
    var suma=0;  
    for(var i=0;i<arguments.length;i++)  
        Sumat=arguments[i];  
    return suma;  
}
```

poziv  
 $\nearrow zbir(2,8)$   
 $zbir(2,8,3,4,5)$

## 13. JS funkcije

U JS-u, kao ulazni argument funkcije može da se javi i neka druga funkcija.

Funcion odredi (funk, p){

```
return function() {
```

```
for(var i=0, i<arguments.length; i++)  
    arguments[i] += p;
```

```
y     return funk.apply(null, arguments)
```

function saberi(a,b,c){

y return a+b+c;

var prva=odradi(Saberi,4);

prva (4,3,7); → ondiko argumentata koliko ima fja  
saber; ili ne?

# 14. Razlike FE i FD

Function Expression

Function Declaration

Deklaracija f je:

```
function square(a){  
    return a*a;  
}
```

Deklaracije su učitane  
pre izvršavanja bilo  
kog koda - HOISTING

Funkcijski izraz:

```
var sg = function(a){  
    return a*a;  
}
```

→ Ne može da se  
potiče PRE  
deklaracije  
(prvo deklaracija, pa  
koriscenje)

```
console.log(square(3))
```

Console.log(sg(3)); → greška ako nakon  
Console.log sledi deklaracija

## 15. HOISTING U JS-U

Kada JS kompajlira sav kod, sve deklaracije varijabli koje koriste ~~var~~<sup>su podignute</sup> na vrh njihovog funkcionalnog / lokalnog opsega (scope-a) (za deklaracije unutar funkcije) ili na vrh njihovog globalnog scope-a (ako su deklarisane izvan funkcija), bez obzira na to gde je stvarna deklaracija napisana. To se zove HOISTING.

Deklaracije funkcija se takođe podignuće, ali one idu na sam vrh, tako da će biti iznad svih deklaracija varijabli.

`x=5; → dodela vrednosti  
alert(x); → izvod  
var x; → deklaracija x`

## 16. LAMBDA f-je

To su anonimne f-je koje se koriste kao podaci (funkcija se prosleđuje kao parametar neke druge f-je, vraća kao povratna vrednost f-je, ili se dodjeljuje promenljivama ili strukturama podataka).

```
const foo = [1, 2, 3];
```

```
const baz = foo.map(function bar(n) { return n+1; });
```

lambda funkcija ali nije  
anonimna

# 17. Closure

Closure predstavlja unutrašnju f-ju koja ima pristup promjenljivama spoljašnje (roditeljske) f-je. Sustina je kad se f-ja poziva, ona osim liste parametara napravi i kontekst tj. spisak promjenljivih koje f-ja koristi, a koje su definisane izvan nje.

Postoji 1 globalni kontekst i više njih za f-je. Svaki poziv f-je kreira novi kontekst.

Closure ima 3 lanca opsega (scope chain)

- ima pristup sopstvenom opsegu (varijablama definisanim unutar {})
- ima pristup promjenljivama roditeljske f-je i parametrima
- ima pristup globalnim promjenljivama

```
function createUser(){
```

```
    let userID = 100,
```

```
    return {
```

```
        getID: function() { return userID; },
```

```
        setID: function(x) { userID = x; }
```

```
    }
```

y

```
var user1 = createUser();
user1.setID(102);
user1.getID();
```

```
function outer() {
    var a = "Hello";
    return function inner() {
        console.log(a);
    }
}
```

y

# 18. IIFE → Immediately Invoked Function Expression (ifi)

IIFE je mehanizam u JS-u koji predstavlja deklaraciju neke funkcije uz njen automatski poziv odmah nakon deklaracije.

```
var nakuadrat = funkcijski izraz (function (stepon) {  
    return function (x) {  
        return Math.pow(x, stepon);  
    }  
});
```

---

```
var pow = function (stepon) {  
    return function (x) { → Pro FE   
        return Math.pow(x, stepon);  
    }  
};
```

```
var nakub = (pow)(3); → IIFE
```

```
console.log(nakub(4)); → output = 43
```