

# DOM

JavaScript - PHP

***ITAcademy***

---

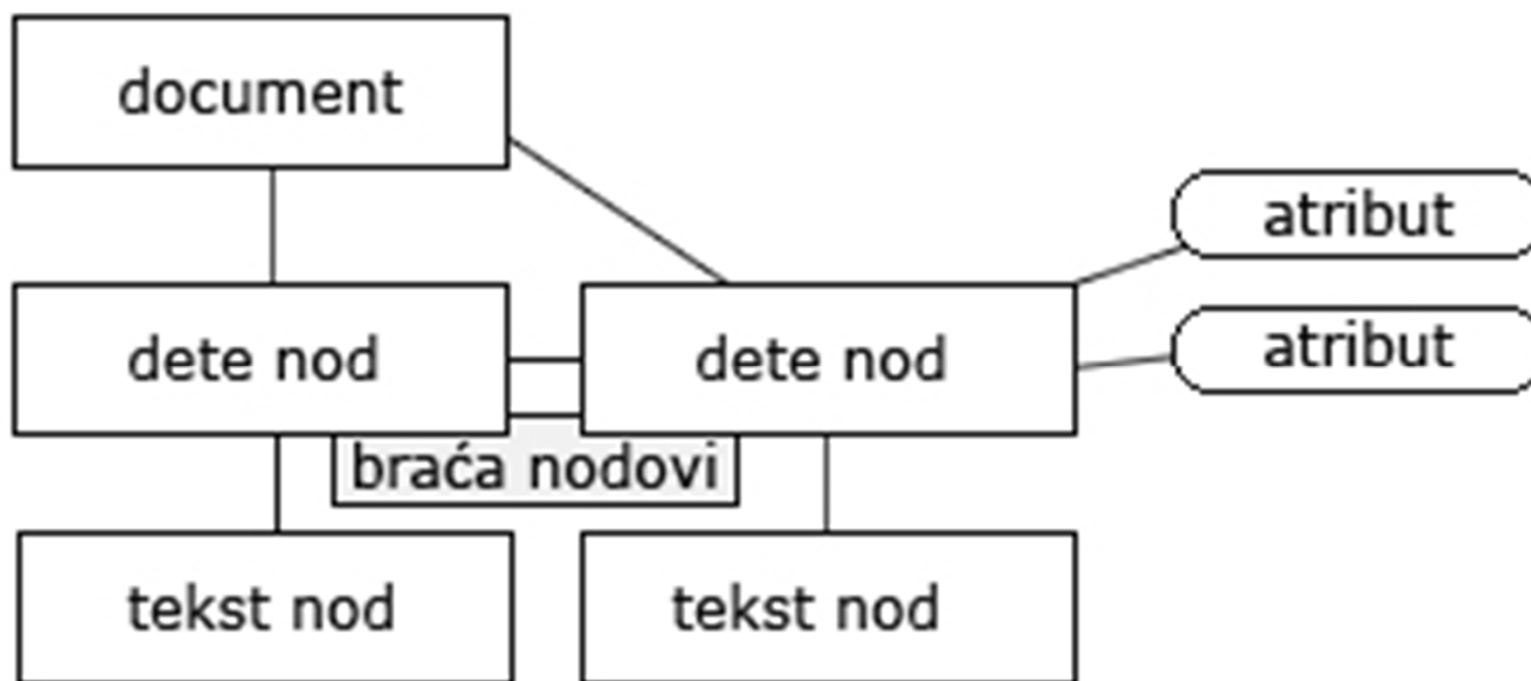
# Šta je DOM

- Document Object Model
- XML dokument pretvoren u objektni model
- Aktivira se nakon parsiranja strane
- U JavaScriptu, kompletna struktura sadržana je u objektu **document**
- [http://www.w3schools.com/jsref/dom\\_obj\\_document.asp](http://www.w3schools.com/jsref/dom_obj_document.asp)

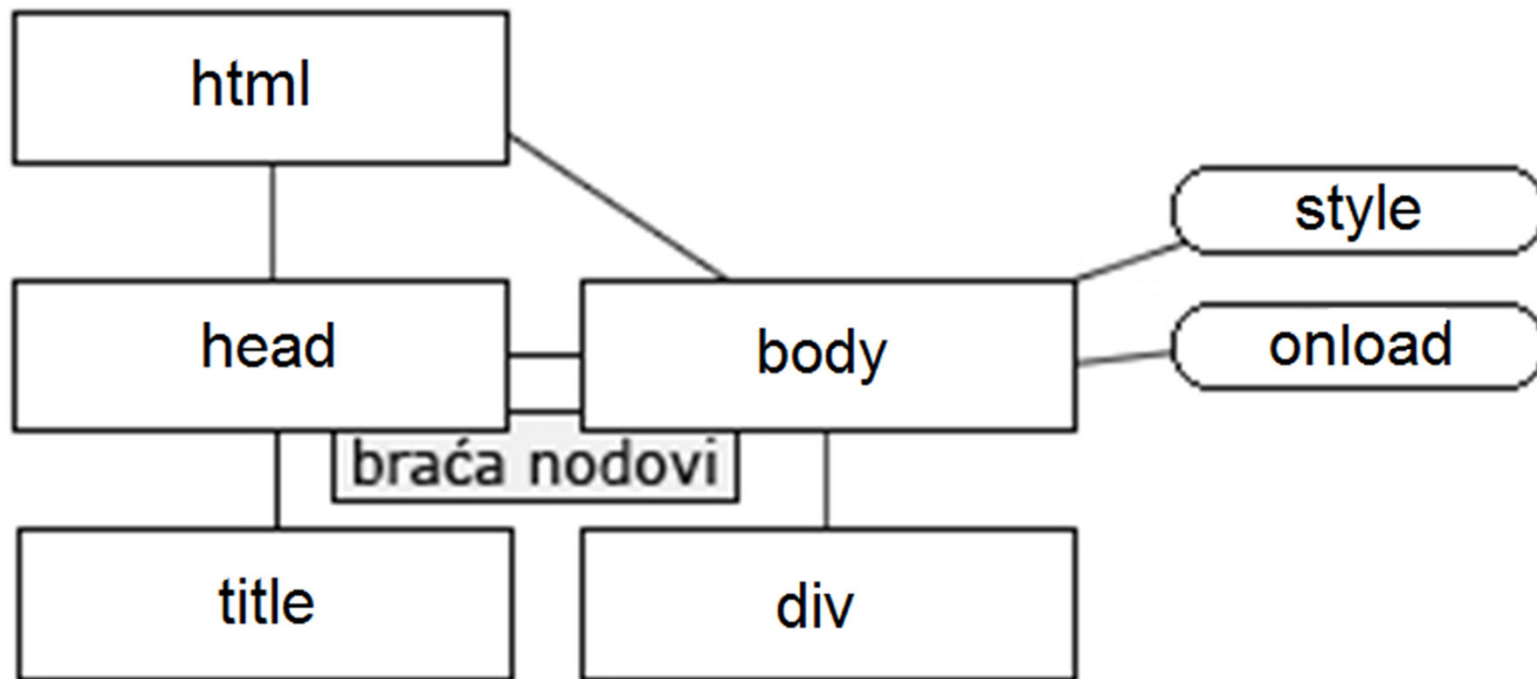
# Struktura DOM-a

- Sadrži nodove, podnodove (decu nodove) i attribute.
- Svaki nod sadrži attribute
- Neki nodovi mogu imati registrovane događaje

# Struktura DOM-a



# Struktura DOM-a



# Rukovanje DOM-om

- DOM metode
- Navigacija u DOM-u
- Manipulacija DOM-om

# DOM metode

- DOM metode su komande (metode) kojima možemo vršiti manipulaciju nad različitim članovima DOM objekta.
  - *createElement('element\_name')*
  - *createTextNode('text content')*
  - *appendChild(element)*
- ...

# Navigacija po dokumentu

- **Pretraga po id-u**  
`document.getElementById("postnavigation");`
- Metod `getElementById` koristimo kada hoćemo da pronađemo element čiji id znamo.
- Id koji tražimo mora biti unikatan za tu stranu



# Pretraga po id-u

- Recimo da imamo dokument sa četiri div elementa

```
<div id="first" style="width:100px; height:100px; background-color:red;">Hello</div>  
<div id="second" style="width:100px; height:100px; background-color:yellow;">From</div>  
<div id="third" style="width:100px; height:100px; background-color:green;">My</div>  
<div id="forth" style="width:100px; height:100px; background-color:blue;">Page</div>
```



- Bilo koji od elemenata možemo identifikovati metodom **getElementById**
- Recimo zatim da hoćemo da identifikujemo zeleni element i smestimo ga u promenljivu g. Napisaćemo:  
***var g = document.getElementById('third');***
- Kada je element jednom identifikovan i smešten u neku promenljivu, možemo manipulirati njime kroz tu promenljivu

```
var g = document.getElementById('third');  
g.innerHTML = "I am green!";
```

# Pretraga po id-u

- Kada radimo sa elementima na strani, veoma je važno da se navigacija vrši nakon iscrtavanja elemenata, a ne pre toga:

```
<div id="first" style="width:100px; height:100px; background-color:red;">Hello</div>
<div id="second" style="width:100px; height:100px; background-color:yellow;">From</div>
<div id="third" style="width:100px; height:100px; background-color:green;">My</div>
<div id="forth" style="width:100px; height:100px; background-color:blue;">Page</div>
<script>
var g = document.getElementById('third');
g.innerHTML = "I am green!";
</script>
```

- Ispravno je i funkcionisaće

```
<script>
var g = document.getElementById('third');
g.innerHTML = "I am green!";
</script>
<div id="first" style="width:100px; height:100px; background-color:red;">Hello</div>
<div id="second" style="width:100px; height:100px; background-color:yellow;">From</div>
<div id="third" style="width:100px; height:100px; background-color:green;">My</div>
<div id="forth" style="width:100px; height:100px; background-color:blue;">Page</div>
```

- Neispravno je i nikada neće funkcionisati

# Navigacija po dokumentu

- Pretraga po nazivu tag-a

`document.getElementsByTagName("div");`

← **Ne zaboravite slovo s**

- Pretražujemo element po nazivu taga (div, head, body...)
- Ova pretraga ima smisla kada nemamo neke druge identifikatore (na primer id), ili kada hoćemo da preuzmemo više elemenata odjednom
- Za razliku od pretrage po id-u, ovaj metod vraća niz elemenata (čak i ako pronade samo jedan)

# Pretraga po nazivu tag-a

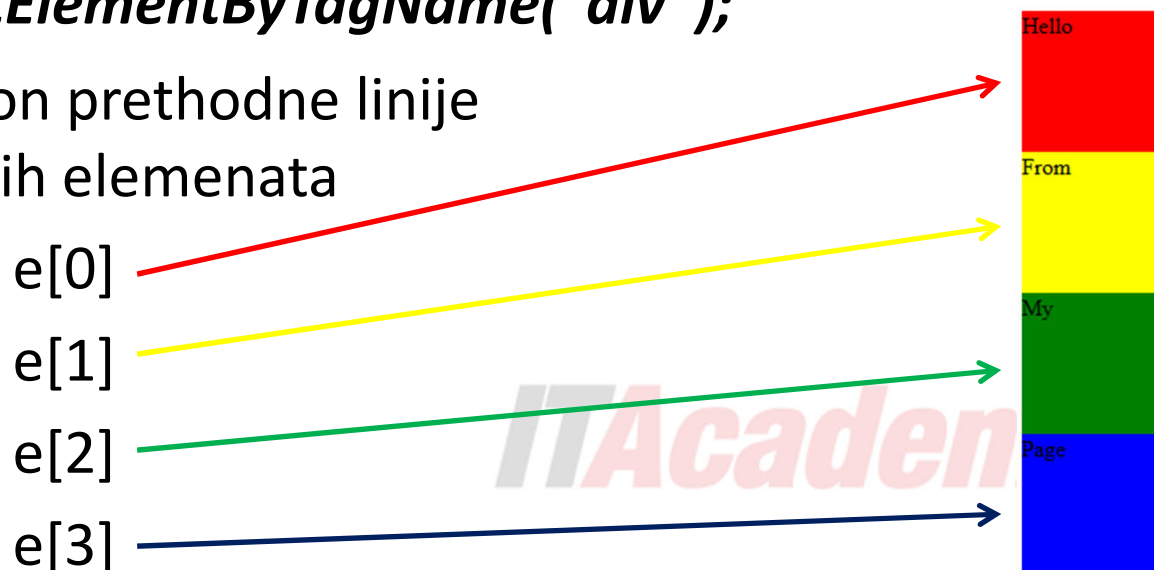
- Pretpostavimo da u prethodnom primeru nemamo id attribute na elementima:

```
<div style="width:100px; height:100px; background-color:red;">Hello</div>  
<div style="width:100px; height:100px; background-color:yellow;">From</div>  
<div style="width:100px; height:100px; background-color:green;">My</div>  
<div style="width:100px; height:100px; background-color:blue;">Page</div>
```

- Da bi došli do nekog od elemenata, morali bi prvi preuzeti sve elemente istog naziva, a zatim identifikovati onaj koji želimo

***var e = Document.getElementsByTagName("div");***

- Promenljiva e, nakon prethodne linije sadrži niz traženih elemenata



# Pretraga po nazivu tag-a

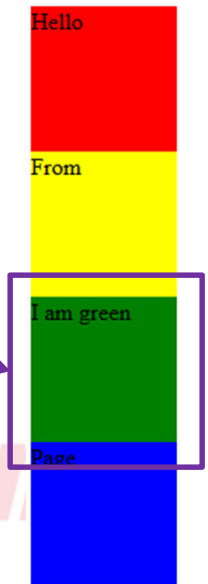
- Nakon primera iz prethodnog slajda, treći (zeleni) element je dostupan u promenljivoj: `e[2]`, pa njegov sadržaj možemo promeniti na sledeći način:

```
var e = document.getElementsByTagName('div');  
e[2].innerHTML = "I am green";
```

# Pretraga po nazivu tag-a

- Nakon primera iz prethodnog slajda, treći (zeleni) element je dostupan u promenljivoj: `e[2]`, pa njegov sadržaj možemo promeniti na sledeći način:

```
var e = document.getElementsByTagName('div');  
e[2].innerHTML = "I am green";
```



# Navigacija po dokumentu

- **Pretraga po vrednosti name atributa**  
`document.getElementsByName("author");`
- **Pretraga po vrednosti atributa class**  
`document.getElementsByClassName("myclass");`
- Za razliku od metode `getElementById`, metode `getElements...` je moguće izvršiti i nad samim elementom, a ne samo na dokumentu

# Navigacija po dokumentu

## Kretanje kroz nodove

- Roditeljski element  
obj.parentNode;
- Deca elementi  
obj.childNodes
- Prvo dete element  
obj.firstChild
- Braća elementi  
obj.nextSibling  
obj.previousSibling



# Zadatak

- Strana ima sledeći sadržaj:

```
<script>
var language = 'en_us';
var locales = {
  'en_us':{
    hello:"Hello",
    welcome:"Welcome to our page",
    how_are_you:"How are you today?"
  },
  'sr_rs':{
    hello:"Dobar dan",
    welcome:"Dobro došli na našu stranu",
    how_are_you:"Kako ste danas?"
  }
}
</script> <div style="font-size:2em; font-weight:bold;">hello</div>
<div style="font-size:1.5em; font-weight:bold;">welcome</div>
<div>how_are_you</div>
```

# Zadatak

- Potrebno je kreirati JavaScript program koji će na osnovu sadržaja div elemenata, na njegovo mesto postavljati odgovarajuću lokalizovanu frazu iz liste fraza reprezentovane pomoću objekta locales.
- Kombinacija fraza preuzima se na osnovu parametra language, koji se nalazi u vrhu strane

# Podsetimo se...

- Svojstvo objekta možemo dobiti pomoću tačke ili uglastih zagrada:

**locales['en\_us']** i **locales.en\_us** daju isti rezultat

- Sve elemente određenog naziva, možemo dobiti pomoću metode **getElementsByTagName**
- Sadržaj elementa menjamo ili preuzimamo svojstvom **innerHTML**

# Rešenje

```
<script>
  var language = 'en_us';
  var locales = {
    'en_us':{
      hello:"Hello",
      welcome:"Welcome to our page",
      how_are_you:"How are you today?"
    },
    'sr_rs':{
      hello:"Dobar dan",
      welcome:"Dobro došli na našu stranu",
      how_are_you:"Kako ste danas?"
    }
  }
</script>
<div style="font-size:2em; font-weight:bold;">hello</div>
<div style="font-size:1.5em; font-weight:bold;">welcome</div>
<div>how_are_you</div>
<script>
  var all_elements = document.getElementsByTagName('div');
  for(var i=0;i<all_elements.length;i++){
    all_elements[i].innerHTML = locales[language][all_elements[i].innerHTML];
  }
</script>
```

# Dodatak

- U rešenju smo dobavili rečnik za traženu lokalizaciju, ali je problem što smo samu lokalizaciju unosili ručno (hard kodirali)
- Bolje bi bilo kada bi lokalizaciju mogli da dobavimo dinamički, na primer, iz url stringa
- U JavaScript-u možemo lako dobiti parametre iz url stringa, sledećim svojstvom:

**document.location.search**

- Ali šta dalje ukoliko želimo da se lokalizacija preuzima iz url parametra **loc**

# Dodatak - pomoć

- U JavaScript-u lako možemo preseći neki string po nekom karakteru i pretvoriti ga u niz
- Na primer, ako imamo string: "hello world" i hoćemo da ga pretvorimo u niz od dve reči (presečen po oznaci space), možemo napisati:

***"hello world".split(' ');***

- Ili

***var w = "hello world";***

***var w\_array = w.split(' ');***

- U JavaScript-u možemo lako eliminisati ili zameniti neki deo stringa nekim drugim. Na primer, ako hoćemo da izbacimo oznaku ? Iz stringa, možemo napisati:

***var s = "hello ? World";***

***var s1 = s.replace("?", "");***

## Dodatak - rešenje

```
var q = document.location.search;
```

```
var q = q.replace("?", "");
```

```
var lang = q.split("=")[1];
```

```
var language = lang;
```

```
var locales = {
```

```
...
```

# Kreiranje objekta u document-u

- Kreiranje objekta (elementa)

***var dv = document.createElement("div");***

- Dodavanje (objekta) elementa dokumentu

***document.body.appendChild(dv);***

- Brisanje elementa

***document.body.removeChild(document.getElementById("abc"));***



# Zadatak

- Aplikacija sadrži sledeći kod:

```
<body></body>
```

```
<script>
```

```
var persons = [
```

```
{
```

```
    id: 1,
```

```
    name: "PADME AMIDALA",
```

```
....
```




Ostatak koda preuzeti sa:

<http://pastie.org/pastes/8653294/text>

(original: <http://starwars.com/explore/encyclopedia/>)

# Zadatak

- Postojeći JavaScript kod treba izmeniti tako da sadržaj strane bude sledeći (bez upotrebe html koda u bilo kom delu programa)

#	name	description	image
1	PADME AMIDALA	An idealist during a time of corruption and war in the Republic, Padmé Amidala was determined to fix what wrongs she could in the Senate, where she represented her idyllic home planet of Naboo. Despite her busy career, Padmé found time for a secret, forbidden marriage with Anakin Skywalker, the Jedi hero of the Clone Wars. Though not a soldier, Padmé nonetheless found herself often in the thick of danger as she courageously defended her deepest held beliefs.	
2	BOBA FETT	A faceless enforcer, Boba Fett's distinctive, customized Mandalorian armor strikes fear in the hearts of fugitives. He is a legendary bounty hunter, accepting warrants from both the Empire and the criminal underworld. He is all business, laconic, and deadly. Fett has carefully guarded his past, cultivating a curtain of mystery around his origins. He is in truth a clone, an exact genetic replica of his highly skilled "father," Jango Fett. From Jango, Boba learned valuable survival and martial skills, and even as a child he was proficient with a blaster or laser cannon. With Jango's death at the hand of Jedi Knight Mace Windu, Boba would forever harbor a deep hatred for the Jedi and their allies.	
3	DARTH	Darth Maul began life as one of the Nightbrothers on Dathomir. Darth Sidious brutally trained his apprentice to be a weapon of the Sith as well as a scheming mastermind. A relentless and acrobatic warrior with an extremely dangerous double-ended lightsaber, Maul longed for revenge.	

# Rešenje

```
var table = document.createElement('table');
table.border = 1;
table.width = 800;

var head_row = document.createElement('tr');
    var id_column = document.createElement('td');
        id_column.innerHTML="#";
    var name_column = document.createElement('td');
        name_column.innerHTML="name";
    var description_column = document.createElement('td');
        description_column.innerHTML="description";
    var image_column = document.createElement('td');
        image_column.innerHTML="image";
    head_row.appendChild(id_column);
    head_row.appendChild(name_column);
    head_row.appendChild(description_column);
    head_row.appendChild(image_column);
table.appendChild(head_row);
document.body.appendChild(table);

for(var i=0;i<persons.length;i++){
    var current_person = persons[i];
    var table_row = document.createElement('tr');
        var id_column = document.createElement('td');
            id_column.innerHTML=current_person.id;
        var name_column = document.createElement('td');
            name_column.innerHTML=current_person.name;
        var description_column = document.createElement('td');
            description_column.innerHTML=current_person.description;
        var image_column = document.createElement('td');
            var imgcol = document.createElement('img');
            imgcol.height = 200;
            imgcol.src = current_person.image;
            image_column.appendChild(imgcol);
        table_row.appendChild(id_column);
        table_row.appendChild(name_column);
        table_row.appendChild(description_column);
        table_row.appendChild(image_column);
    table.appendChild(table_row);
}
```

<http://pastie.org/pastes/8653343/text>

Komentarisati loše strane  
ovog rešenja

# Zadatak

- Od postojećih slika (iz prehtodnog primera), napraviti jednostavan slide show.
- Inicijalni podaci:
- `var images =`  
`["http://starwars.com/img/explore/encyclopedia/characters/padmeamidala_detail.png", "http://starwars.com/img/explore/encyclopedia/characters/bobafett_detail.png", "http://starwars.com/img/explore/encyclopedia/characters/darthmaul_detail_c.png", "http://starwars.com/img/explore/encyclopedia/characters/yoda_detail.png"];`

# Rešenje

...

```
var image = document.createElement("img");  
document.body.appendChild(image);  
var current_image = 0;  
setInterval("image.src=images[current_image++]; if(current_image>=images.length)  
current_image=0;", 3000);
```

...

Pokušajte da optimizujete program tako da svi podaci aplikacije budu izolovani od globalnog konteksta (da budu unutar zasebnog objekta)

# Stilizacija objekata u DOM-u

- Svi stilski atributi se nalaze u svojstvu style
- Koristi se ista notacija kao i u CSS-u, osim što se ne koristi oznaka - već camel case notacija

***domObjekat.style.position="absolute";***

***domObjekat.style.top="100px";***

***domObjekat.style.left="100px";***

***domObjekat.style.backgroundColor="black";***

***domObjekat.style.zIndex=500;***

***domObjekat.style.width="100px";***

***domObjekat.style.height="100px";***

- Takođe može i:

***domObjekat.style.cssText="width:100px;height:200px; ...";***

# HTML Događaji

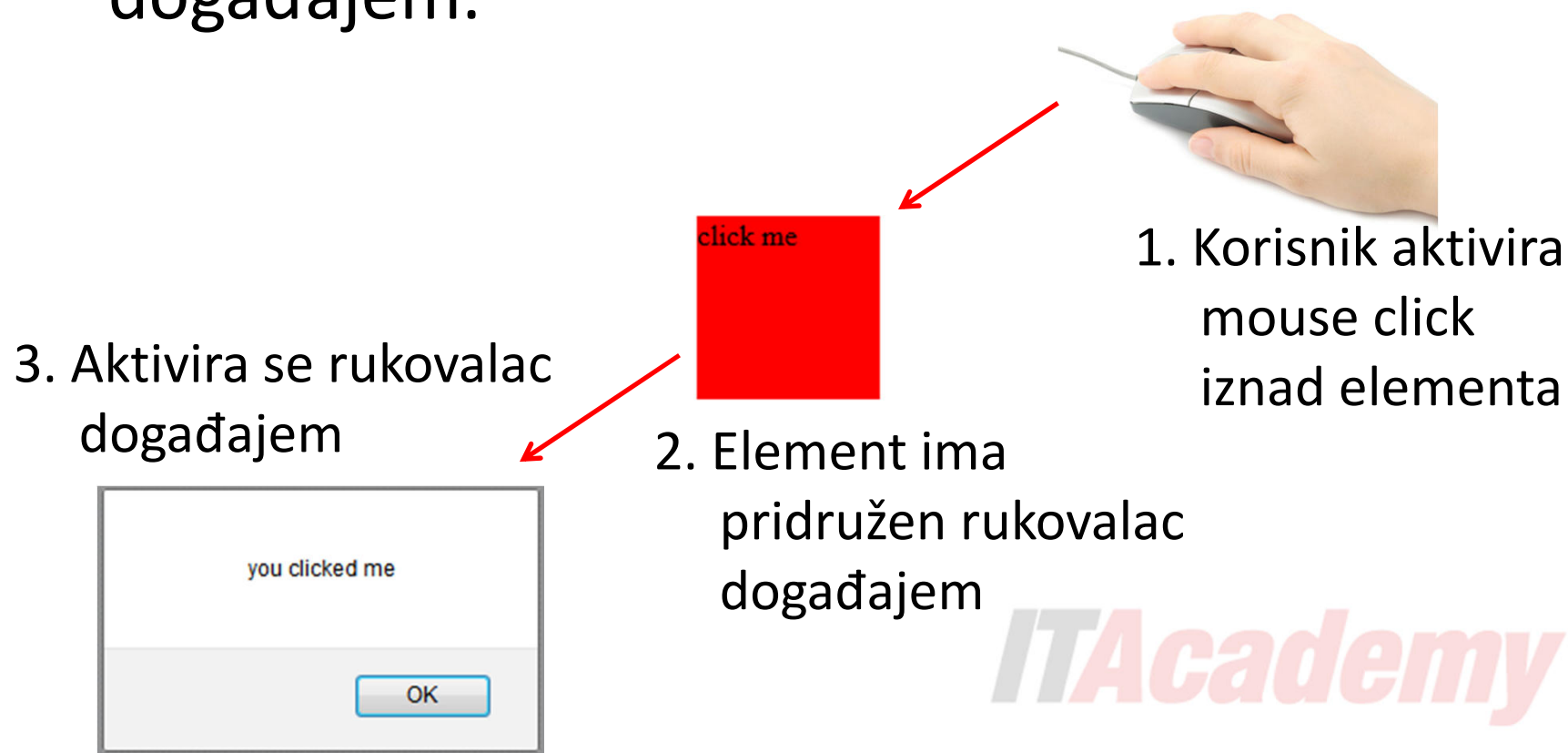
[http://www.w3schools.com/tags/ref\\_eventattributes.asp](http://www.w3schools.com/tags/ref_eventattributes.asp)

- Svi ili gotovo svi elementi u HTML-u u stanju su da reaguju na neke događaje. Ovi događaji su najčešće vezani za korisnički interfejs i tiču se miša ili tastature, ali takođe i nekih elementarnih pojava na strani (skrolovanja, napuštanja kontrole i slično)
- Događaje možemo “uhvatiti” kombinacijom atributa nad elementom čiji nas događaji interesuju i koda koji će taj događaj obraditi.



# HTML Događaji

- Kod koji se aktivira kao rezultat aktivacije nekog događaja, naziva se rukovalac događajem.





# HTML Događaji

- Rukovaoci događajem mogu biti definisani deklarativno, unutar html taga, pomoću html atributa.
- U tom slučaju, html atribut sadrži JavaScript kod koji će biti aktiviran prilikom aktivacije događaja
- Događaj sa prethodnog slajda mogao bi biti realizovan na sledeći način

Rukovalac događajem **onclick**

```
<body>
<div
  onclick="alert('you clicked me')" id="container"
  style="width:100px;height:100px;background-color:red;"
>click me</div>
</body>
```

my

# Koje sve događaje možemo uhvatiti

- Postoji mnoštvo događaja koje možemo uhvatiti, ali njihov tačan broj se ne može odrediti, jer ne postoje svi događaji na svim browserima.
- Ipak, većina standardnih događaja postoji u svim browserima, i obrađuje se na isti ili veoma sličan način
- Najčešće obrađivani događaji su:
  - Događaji miša (onclick, onmouseover, onmousedown, onmouseup, onmouseenter)
  - Događaji tastature (onkeyup, onkeydown, onkeypress)
  - Događaji dokumenta i prozora (onload, onunload, onresize)
  - Događaji kontrola (onblur, onfocus, onchange)

<http://www.quirksmode.org/dom/events/index.html>

# Primer

- Potrebno je kreirati dva div elementa
- Prilikom klika mišem na jedan div element, njegov sadržaj se kopira u drugi div element

# Primer

- Potrebno je kreirati dva div elementa
- Prilikom klika mišem na jedan div element, njegov sadržaj se kopira u drugi div element
- **Rešenje:**

```
<div onclick="document.getElementById('target_div').innerHTML  
= this.innerHTML">
```

*click me*

```
</div><div id="target_div"></div>
```

- Pokušajte da modifikujete primer tako da prelaskom mišem preko ciljnog elementa obriše njegov sadržaj

# Programabilna dodela slušača događaja

- Osim pomoću html tagova, slušače događaja je moguće pridružiti nekom dom elementu i programabilno.
- Tada svojstvu kojim je predstavljen događaj pridružujemo odgovarajuću funkciju

```
var dv = document.createElement("div");  
dv.innerHTML = "click me";  
dv.onclick = function() { alert('you clicked me!'); };  
document.body.appendChild(dv);
```

# Programabilna dodela slušača događaja

- Osim direktnog pridruživanja pomoću atributa, rukovaoca događajem je moguće pridružiti elementu i pomoću metode **addEventListener**

```
var dv = document.createElement("div");  
dv.innerHTML = "click me";  
dv.addEventListener("click",function(){ alert('you clicked me!'); });  
dv.addEventListener("click",function(){ alert('indeed!'); });  
document.body.appendChild(dv);
```

# Argumenti događaja

- Prilikom aktivacije, događaji rukovaocu prosleđuju i određene informacije kroz parametar funkcije
- Ove informacije su karakteristične za događaj i obično sadrže baš informacije koje su nam potrebne a vezane su za događaj
- Na primer, ako je u pitanju događaj aktivacije miša, dobićemo koordinate na kojima se događaj aktivirao i taster koji je pritisnut

# Argumenti događaja

```
var dv = document.createElement("div");  
dv.innerHTML = "click me";  
dv.onclick = function(event){  
    console.debug(event);  
};  
document.body.appendChild(dv);
```



# Primer

Treba kreirati program koji će omogućiti korisniku da crta po div tagu pomoću događaja mousedown i mouseup

<http://pastie.org/pastes/8674673/text>

# Rešenje

```
var dv = document.createElement("div");
dv.style.cssText = "width:500px; height:500px; border:1px solid
    #d1d1d1;";
var mousedown = false;
dv.onmousedown = function(){ mousedown = true; };
dv.onmouseup = function() { mousedown = false; };
dv.onmousemove = function(event) {
    if(!mousedown) return;
    var pt = document.createElement("div"); pt.style.cssText =
        "position:absolute;background-color:red;width:5px;height:5px;";
    pt.style.left = event.clientX+"px";
    pt.style.top = event.clientY+"px";
    dv.appendChild(pt);
}
document.body.appendChild(dv);
```

# Zadatak

- Treba napraviti sistem koji će vršiti validaciju tekst kontrole, tako da u nju mogu da budu uneti samo brojevi
- Ukoliko korisnik u nju unese bilo šta što nije broj, kontrola automatski dobija crveni okvir

# Rešenje

```
<body>
```

```
<input type="text" onkeyup="checkControl(this)"  
  placeholder="Enter number" />
```

```
</body>
```

```
<script>
```

```
function checkControl(control){  
    control.style.border=(isNaN(control.value))?"1px solid  
    red":"";  
}
```

```
</script>
```

# Zadatak

- Napraviti funkciju \$
- Ova funkcija prihvata jedan parametar (id elementa), a kao rezultat vraća element sa prosleđenim id-om
- Izmeniti strukturu elementa uz pomoć napravljene funkcije i innerHTML svojstva

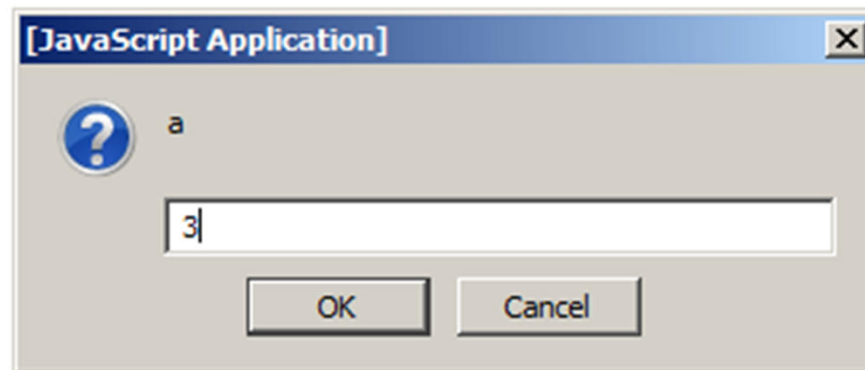
# Rešenje

```
<script>  
var $ = function(o)  
{  
    return document.getElementById(o);  
}  
</script>  
<div id="myId"></div>
```

# Zadatak

- Kreirati aplikaciju koja sadrži jedan div tag.
- Kada korisnik aktivira levi taster miša nad tagom, otvaraju se dva boksa za unos a i b promenljive
- Kada korisnik unese obe promenljive, div tagu se pridružuje div tag koji sadrži rezultat sabiranja dva uneta broja (promenljive)

ADD



ADD

5

9

11

# Rešenje

```
<script>
function addNode(o)
{
    var d = document.createElement("div");
    d.innerHTML=(parseInt(prompt("a"))+parseInt(prompt("b")));
    o.appendChild(d);
}
</script>
<div onclick="addNode(this)" style="cursor:pointer;">
ADD
</div>
```



# Zadatak

- Potrebno je kreirati arr objekat (asocijativni niz)
- Potrebno je kreirati div tag koji sadrži tekst SET MENU
- Kada se nad ovim tagom aktivira levi taster miša izgrađuje se meni na osnovu podataka iz niza

SET MENU

google

yahoo

bing

# Rešenje

```
<script>
var arr =
    {"google":"http://www.google.com","yahoo":"http://www.yahoo.com","bing":
    "http://www.bing.com"};
function createMenu(o)
{
    var c = document.getElementById(o);
    for(var i in arr)
    {
        var a = document.createElement("a");
        a.innerHTML=i;
        a.href=arr[i];
        a.style.marginRight="20px";
        a.style.border="1px solid red";
        a.style.padding="5px";
        c.appendChild(a);
    }
}
</script>
<div onclick="createMenu('menuCont')" style="cursor:pointer;">SET MENU</div>
<div id="menuCont"></div>
```

# Zadatak

- Kreirati niz koji sadrži listu slika (slike se nalaze u istom folderu gde i strana)
- Kreirati funkciju switchImage koja na osnovu niza emituje slike u intervalu od nekoliko sekundi
- Kada se završi emitovanje slika, potrebno je započeti ciklus od početka

# Rešenje

```
<script>
var arr =
    ["Chrysanthemum.jpg","Desert.jpg","Hydrangeas.jpg","Jellyfish.jpg","Koala.jpg","Lighthouse.jpg","Pen
    guins.jpg","Tulips.jpg"];
var allImages = arr.length;
var actImage=0;
function switchImage()
{
    var img = document.getElementById("imgPh");
    img.src=arr[actImage];
    if(actImage<allImages-1)
        actImage++;
    else
        actImage=0;
    setTimeout("switchImage()",1000);
}
</script>
<div id="menuCont">
    <img id="imgPh" src="" />
</div>
```

# Zadatak

- Kreirati program koji će pretražiti kompletnu stranu i ako unutar nekog od tagova pronade odgovarajući string, obojice taj tag u specifičnu boju
- Pomoć:
  - Metodom `indexOf` može se proveriti postojanje određenog stringa unutar drugog stringa:
  - *`indexOf("somesstring")`*

## Rešenje:

```
var allDivs =  
    document.getElementsByTagName("p");  
for(var i=0;i<allDivs.length;i++){  
    if(allDivs[i].innerHTML.indexOf("timestamp")  
        >0)  
        allDivs[i].style.backgroundColor="red";  
}
```

# Zadatok:

- Kreirati Padajući meni

Google Yahoo Bing  
Bing 1  
Bing 2  
Bing 3

# HTML kod

```
<body>
<ul id="meni">
  <li id="stavka">
    <a href="#">Google</a>
    <ul>
      <li><a href="#">Google 1</a></li>
      <li><a href="#">Google 2</a></li>
      <li><a href="#">Google 3</a></li>
    </ul>
  </li>
  <li id="stavka">
    <a href="#">Yahoo</a>
    <ul>
      <li><a href="#">Yahoo 1</a></li>
      <li><a href="#">Yahoo 2</a></li>
      <li><a href="#">Yahoo 3</a></li>
    </ul>
  </li>
  <li id="stavka">
    <a href="#">Bing</a>
    <ul>
      <li><a href="#">Bing 1</a></li>
      <li><a href="#">Bing 2</a></li>
      <li><a href="#">Bing 3</a></li>
    </ul>
  </li>
</ul>
</body>
```



# CSS

```
<style>
#meni{
  padding: 0px;
}
ul {
list-style-type: none;
}
#meni li {
  margin-right: 6px;
  float: left;
}
#meni li ul
{
  padding: 0px;
  background-color: #d3d3d3;
}
#meni li ul li
{
  padding: 4px;
  background-color: #d3d3d3;
  clear: both;
}
</style>
```

# JavaScript

```
var meni = document.getElementById("meni");
var sveStavke = meni.getElementsByTagName("li");
for(var i=0;i<sveStavke.length;i++)
{
    if(sveStavke[i].id=="stavka") {
        var podstavke = sveStavke[i].getElementsByTagName("ul")[0];
        podstavke.style.display = 'none';
        sveStavke[i].onmouseout = function(){
            var podstavke = this.getElementsByTagName("ul")[0];
            podstavke.style.display = 'none';
        }
        sveStavke[i].onmouseover = function() {
            var podstavke = this.getElementsByTagName("ul")[0];
            podstavke.style.display = "";
            podstavke.onmouseout = function() {
                this.style.display = 'none';
            }
        }
    }
}
```