Rukovanje prozorom web pregledača

Poznavanje sintaksnih i leksičkih pravila JavaScript jezika danas je od presudne važnosti za uspešno bavljenje poslom frontend programera. Ipak, odlično poznavanje osnovnih leksičkih elemenata jezika (tipova, operatora, promenljivih, petlji, grananja, objekata...) samo je ulaznica za svet klijentskog programiranja na webu. Drugim rečima, pored dobrog poznavanja JavaScripta, za kreiranje modernih sajtova neophodno je i razumevanje funkcionalnosti koje web pregledači stavljaju na raspolaganje programerima. Bez takvih gotovih funkcionalnosti, JavaScript bi bilo jezik čiju moć bi bilo nemoguće primetiti.

Tako web pregledači poseduju ugrađene mehanizme za kontrolu ponašanja prozora unutar koga se nalazi stranica web sajta, funkcionalnosti za kontrolu sadržaja, strukture i stilizacije, mehanizme za kontrolisanje istorije posećenih stranica, upućivanje zahteva serverima i još mnogo toga. Opisani mehanizmi, na primer, omogućavaju da se JavaScript iskoristi za dinamičko ažuriranje sadržaja nekog HTML elementa. Stoga će modul pred vama biti posvećen osnovnim postulatima korišćenja programske logike koju nam na raspolaganje stavljaju web pregledači.

Coffee Shop web sajt

Pristupi koji budu obrađeni u modulu koji je pred vama omogućiće nam da unesemo dodatnu interaktivnost u sajt koji se razvija u ovom kursu. Nakon definisanja osnovne strukture i stilizacije u lekcijama o korišćenju Bootstrapa, ovaj modul će nam omogućiti da preuzmemo programabilnu kontrolu nad HTML dokumentom i tako dodatno unapredimo sajt koji razvijamo.

Web APIs

Skupovi funkcionalnosti koje web pregledači stavljaju na raspolaganje programeru drugačije se nazivaju aplikativni programski interfejsi weba, odnosno skraćeno Web APIs. API je pojam koji nije striktno vezan za JavaScript ili frontend razvoj, već je reč o nazivu koji se koristi da označi skup gotovih funkcionalnosti bilo koje platforme koje se mogu koristiti za jednostavniju realizaciju kompleksnih zadataka.

Moderni web pregledači izlažu veliki broj različitih API-a koje je moguće koristiti prilikom klijentskog programiranja na webu. Neki od najpoznatijih API-a omogućavaju:

- manipulaciju sadržajem, strukturom i stilizacijom HTML dokumenta;
- slanje pozadinskih zahteva serveru i dobijanje HTTP odgovora, što je osnova za funkcionisanje modernih web aplikacija;
- crtanje vektorske, 2D i 3D grafike unutar HTML dokumenta;
- rukovanje audio i video sadržajem;
- manipulaciju uređajem koji se koristi za pregled web sajta;
- upravljanje lokalnim skladištima podataka.

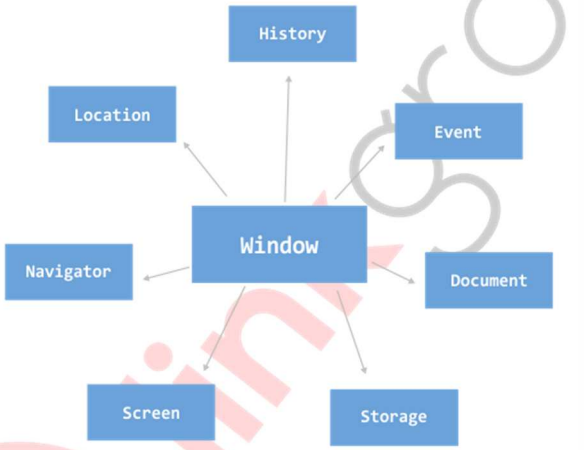
U ovom kursu nas prevashodno interesuje skup funkcionalnosti koje web pregledači izlažu kako bi se omogućila laka kontrola prozora i sadržaja koji se unutar takvog prozora prikazuje.

Objekat Window

Proces prikazivanja neke web stranice unutar web pregledača podrazumeva i kreiranje specijalnog objekta – `Window`. Reč je o objektu koji kreiraju web pregledači i unutar njega tom prilikom smeštaju brojne funkcionalnosti i svojstva koja se mogu koristiti za interakciju sa prozorom web pregledača ili sa samim dokumentom koji je unutar takvog prozora prikazan. Objekat `Window` je zapravo način na koji web pregledači stranici izlažu različite funkcionalnosti koje se mogu koristiti prilikom funkcionisanja web sajta.

Gotovo svi današnji web pregledači poseduju funkcionalnost tabova, odnosno pregleda web stranica u izdvojenim prozorima u sklopu glavnog prozora. U takvoj situaciji, za svaki pojedinačni tab web pregledača obavlja se kreiranje po jednog posebnog `Window` objekta.

Objekat `Window` unutar sebe objedinjuje brojne druge objekte, pri čemu svi takvi objekti nisu striktno vezani za grafičku reprezentaciju prozora i HTML dokumenta. Tako se objekat `Window` može doživeti kao neki omotač za funkcionalnosti različitih Web API-a.



Slika 7.1. Najznačajniji objekti i API-i kojima se može pristupiti korišćenjem Window objekta

Slika 7.1. ilustruje neke od najznačajnijih objekata koji su korišćenjem svojstava `Window` objekta izloženi na korišćenje programskom kodu JavaScript jezika.

BOM – Browser Object Model

Skup određenih funkcionalnosti koje se nalaze unutar `Window` objekta, a namenjene su interakciji sa samim web pregledačem, odnosno prozorom unutar koga se prikazuje HTML dokument, veoma često se kolektivno naziva *Browser Object Model*. Zvanična specifikacija kojom se standardizuje BOM ne postoji, ali većina web pregledača obezbeđuje identične funkcionalnosti. Neki od primera funkcionalnosti koje se svrstavaju u objektni model web pregledača su: mogućnost pomeranja prozora, mogućnost promene veličine prozora, dobijanje informacija o web pregledaču, rukovanje istorijom posećenih web stranica...

Već je rečeno da `Window` objekat automatski kreira web pregledač prilikom parsiranja i prikazivanja stranice. Takvom `Window` objektu može se pristupiti korišćenjem promenljive **window**:

```
window.alert("Hello World!");
```

Prikazana linija koda ilustruje pozivanje `alert()` metode `window` objekta. Metodi se prosleđuje jedan parametar, koji predstavlja tekst koji će biti prikazan kao poruka korisniku unutar modalnog prozora web pregledača.

Verovatno vam je poznato da se upravo prikazana metoda `alert()` može pozvati i bez korišćenja `Window` objekta, jednostavnim navođenjem:

```
alert("Hello World!");
```

I upravo prikazana linija potpuno je legitimna i stvoriće identičan efekat kao i linija u kojoj se metoda `alert()` poziva nad promenljivom `window` koja predstavlja `Window` objekat. Zašto je to tako?

`Window` objekat je zapravo **globalni objekat** unutar web pregledača. To je objekat koji uvek postoji u okviru globalnog sklopa. Upravo zbog toga se za pristup njegovim svojstvima i metodama ne mora navoditi naziv promenljive `window`, pa je to na kraju i razlog zbog kog obe upravo prikazane linije JavaScript koda proizvode identičan efekat.

Evo još jednog primera koji ilustruje upravo opisanu osobinu `Window` objekta. Sa konzolom web pregledača moguće je vršiti interakciju na sledeći način:

```
console.log("Hello World!");
```

Ipak, s obzirom na to da je `console` zapravo jedno od svojstava `Window` objekta, identično je moguće postići i na sledeći način:

```
window.console.log("Hello World!");
```

Pored svojstava i objekata koji su automatski ugrađeni u `Window` objekat, bitno je znati da i kod koji samostalno pišemo automatski postaje deo `Window` objekta:

```
var myVariable = "Hello World!";
console.log(window.myVariable);
```

Prikazanim kodom prvo se kreira jedna promenljiva sa nazivom `myVariable`. Ovakva promenljiva automatski postaje deo globalnog `window` objekta. To se proverava unutar druge linije prikazanog koda, ispisivanjem vrednosti svojstva `myVariable`, objekta `window`. Unutar konzole se dobija vrednost koju smo samostalno definisali, što je dokaz da je promenljiva `myVariable` postala deo `window` objekta.

Identična je situacija i sa korisnički definisanim funkcijama:

```
function sayHello() {
    console.log("Hello World!");
}
```

Ovo je jedna funkcija sa nazivom `sayHello()`. Pored standardnog pozivanja, ona se može pozvati i kao metoda `window` objekta:

```
window.sayHello();
```

Prikazanom linijom, unutar konzole se dobija poruka iz metode `sayHello()`, što je dokaz da i funkcije postaju metode globalnog `window` objekta.

Na kraju priče o `Window` objektu, biće prikazana najznačajnija svojstva i metode koje se nalaze unutar ovog objekta (tabele 7.1 i 7.2).

Svojstvo	Opis
`console`	omogućava pristup `Console` objektu za interakciju sa konzolom web pregledača
`document`	omogućava pristup objektu `Document`, koji obezbeđuje interakciju sa strukturom, sadržajem i stilizacijom dokumenta
`history`	omogućava pristup objektu `History`, za rukovanje istorijom posećenih web adresa
`innerHeight`	visina vidnog polja prozora uključujući trake za skrolovanje
`innerWidth`	širina vidnog polja prozora uključujući trake za skrolovanje
`location`	omogućava pristup `Location` objektu koji dozvoljava uvid u informacije o trenutnoj adresi web sajta
`name`	svojstvo koje predstavlja naziv prozora
`navigator`	obezbeđuje pristup `Navigator` objektu, koji omogućava dobijanje različitih informacija o web pregledaču
`screen`	omogućava pristup `Screen` objektu za dobijanje informacija o displeju korisnika

Tabela 7.1. Najznačajnija svojstva Window objekta

Metoda	Opis
`alert()`	prikazuje modalni prozor sa porukom i OK dugmetom
`blur()`	uklanja fokus sa trenutnog prozora/taba
`close()`	zatvara tekući prozor/tab
`confirm()`	prikazuje modalni prozor sa porukom i dugmićima OK i Cancel
`focus()`	postavlja fokus na tekući prozor/tab
`moveBy()`	pomera prozor u odnosu na trenutnu poziciju
`open()`	otvara novi prozor/tab
`print()`	šalje dokument iz tekućeg prozora na štampu
`prompt()`	prikazuje modalni prozor sa poljem u koje korisnik može da unese neku vrednost
`resizeTo()`	postavlja veličinu prozora na definisanu visinu i širinu
`scrollTo()`	skroluje dokument
`stop()`	zaustavlja aktivno učitavanje prozora

Tabela 7.2. Najznačajnije metode Window objekta

U nastavku ove lekcije biće prikazano nekoliko najinteresantnijih primera korišćenja ugrađenih funkcionalnosti za kontrolisanje web pregledača. Tako će nastavak ove lekcije biti posvećen objektnom modelu web pregledača, odnosno BOM-u. Naredna lekcija biće posvećena još jednom objektnom modelu – objektnom modelu za kontrolisanje dokumenta – *Document Object Model* (skraćeno DOM). Nakon DOM-a, u ovom modulu će biti obrađen i pojam događaja koje web pregledač stavlja na raspolaganje JavaScript kodu HTML dokumenta.

Pitanje

Osnovni objekat unutar koga se objedinjuju sve funkcionalnosti koje web pregledači stavljaju na raspolaganje HTML dokumentu je:

- **Window**
- Object
- Document
- Screen

Objašnjenje:

Objekat Window je način na koji web pregledači stranici izlažu različite funkcionalnosti koje se mogu koristiti prilikom funkcionisanja web sajta.

Dobijanje informacija o displeju korisničkog uređaja

Unutar `Window` objekta nalazi se objekat `Screen`, koji omogućava pristup osnovnim informacijama o displeju korisničkog uređaja. Pristupanje `Screen` objektu se može obaviti na sledeći način: `window.screen`. Kao i u primerima do sada, prefiks `window` je moguće izostaviti, zato što je `Window` globalni objekat web pregledača.

`Screen` objekat izlaže nekoliko veoma korisnih svojstava (tabela 7.3).

Svojstvo	Opis
`screen.width`	širina displeja korisničkog uređaja u pikselima
`screen.height`	visina displeja korisničkog uređaja u pikselima
`screen.availWidth`	širina displeja korisničkog uređaja u pikselima umanjena za određene menije operativnog sistema, ukoliko postoje
`screen.availHeight`	visina displeja korisničkog uređaja u pikselima umanjena za određene menije operativnog sistema, ukoliko postoje
`screen.colorDepth`	osobina displeja korisničkog uređaja koja oslikava broj bitova za predstavljanje boja (16, 24, 32 bita...)

Tabela 7.3. Svojstva Screen objekta

Primer korišćenja upravo prikazanih svojstava može izgledati ovako:

```
let screenWidth = screen.width;
let screenHeight = screen.height;
let availableWidth = screen.availWidth;
let availableHeight = screen.availHeight;
let colorDepth = screen.colorDepth;


console.log("Screen width: " + screenWidth + "\n" +
    "Screen height: " + screenHeight + "\n" +
    "Available screen width: " + availableWidth + "\n" +
    "Available screen height: " + availableHeight + "\n" +
    "Color depth: " + colorDepth + "\n");
```

U zavisnosti od osobina displeja korisničkog uređaja, unutar konzole mogu biti dobijeni različiti podaci:

```
Screen width: 1600
Screen height: 900
Available screen width: 1600
Available screen height: 870
Color depth: 24
```

Prikazani podaci su dobijeni na laptop kompjuteru sa Windows operativnim sistemom. Zanimljivo je primetiti da visina displeja i dostupna visina nisu jednake, upravo zbog postojanja bara, koji je uvek vidljiv na dnu displeja.

Rukovanje URL-om

Web pregledači omogućavaju dobijanje informacija o URL adresi na kojoj se nalazi HTML dokument koji se prikazuje. Za dobijanje takvih informacija koristi se objekat `Location`, koji je sastavni deo globalnog objekta `Window`.

`Location` objektu se može pristupiti navođenjem promenljive `window`, ali i bez nje:

- `window.location`
- `location`

`Location` objekat poseduje nekoliko svojstava, koja omogućavaju dobijanje različitih informacija o URL adresi (tabela 7.4).

Svojstvo	Opis
`location.href`	URL na kome se nalazi prikazani dokument
`location.hostname`	domensko ime sajta
`location.pathname`	putanja i naziv fajla prikazanog dokumenta
`location.protocol`	web protokol (http, https, file...)

Tabela 7.4. Svojstva Location objekta

Svojstva prikazana tabelom 7.4. mogu se upotrebiti na sledeći način:

```
let href = location.href;
let hostname = location.hostname;
let pathname = location.pathname;
let protocol = location.protocol;

console.log("Href: " + href + "\n" +
    "Hostname: " + hostname + "\n" +
    "Pathname: " + pathname + "\n" +
    "Protocol: " + protocol + "\n");
```

U zavisnosti od toga gde se nalazi dokument sa upravo prikazanim JavaScript kodom, unutar konzole se može dobiti različit ispis. On može izgledati ovako:

```
Href: https://www.link-group.eu/index.html
Hostname: www.link-group.eu
Pathname: /index.html
Protocol: https:
```

Imajte na umu da će, ukoliko ovakav kod isprobate unutar dokumenta smeštenog na fajl sistemu lokalnog kompjutera, vrednost `hostname` svojstva biti prazna. Pored toga, u takvoj situaciji će vrednost `protocol` svojstva biti `file`.

Modalni prozori

`Window` objekat poseduje i tri specijalne metode koje omogućavaju kreiranje modalnih prozora, različitih namena (tabela 7.5).

Metoda	Opis
`alert()`	prikazuje modalni prozor sa porukom i OK dugmetom
`confirm()`	prikazuje modalni prozor sa porukom i OK i Cancel dugmićima
`prompt()`	prikazuje modalni prozor sa poljem za unos vrednosti i dugmićima OK i Cancel

Tabela 7.5. Metode za prikaz modalnih prozora

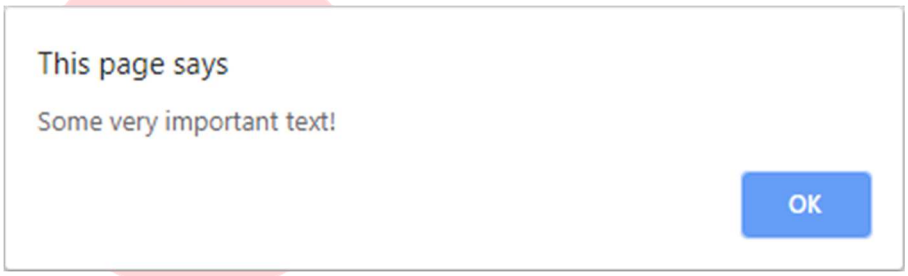
alert()

Modalni prozor koji se dobija metodom `alert()` može se koristiti za prikaz važnih poruka, odnosno poruka za koje želimo biti sigurni da će ih korisnik videti:

```
window.alert("Some very important text!");
```

Naravno, metodu `alert()`, baš kao i metode `confirm()` i `prompt()`, moguće je pozivati bez `window` prefiksa:

```
alert("Some very important text!");
```

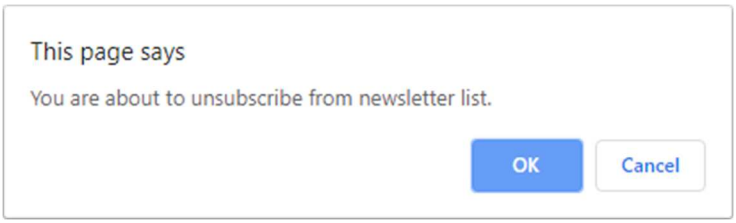


Slika 7.2. Modalni prozor koji se dobija metodom alert()

confirm()

Ukoliko je potrebno omogućiti korisniku da nešto verifikuje ili prihvati, moguće je koristiti metodu `confirm()`:

```
window.confirm("You are about to unsubscribe from newsletter list.");
```



Slika 7.3. Modalni prozor koji se dobija metodom confirm()

Metoda `confirm()` nema mnogo smisla ukoliko se na neki način ne dobije povratna informacija o odabiru koji je napravio korisnik. Stoga ova metoda emituje povratnu vrednost `boolean` tipa, u zavisnosti od toga da li je korisnik odabrao *OK* ili *Cancel* dugme:

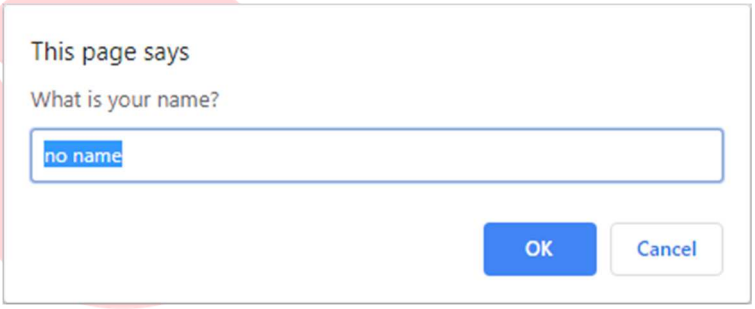
```
            var userConfirmed = window.confirm("You are about to unsubscribe
from newsletter list.");

            if(userConfirmed){
                alert("You are unsubscribed.");
            } else {
                alert("You are not going to be unsubscribed.");
            }
```

Sada je povratna vrednost metode `confirm()` smeštena unutar promenljive `userConfirmed`. Takva promenljiva je zatim iskorišćena kao uslov uslovne naredbe `if...else`. U zavisnosti od korisničkog odabira, prikazuje se odgovarajuća poruka.

prompt()

Kada je od korisnika potrebno preuzeti neku vrednost, može se koristiti metoda `prompt()`:

```
window.prompt("What is your name?", "no name");
```



Slika 7.4. Modalni prozor koji se dobija metodom prompt()

Metoda `prompt()` može da prihvati dva parametra – tekst koji će biti ispisan unutar modalnog prozora i podrazumevanu vrednost koja se automatski pojavljuje unutar polja za unos vrednosti.

Vrednost koju korisnik unese u polje za unos vrednosti moguće je dobiti kao povratnu vrednost metode `prompt()`. Uneta vrednost se isporučuje samo ukoliko korisnik klikne na dugme OK. Kada korisnik odabere dugme Cancel, povratna vrednost metode `prompt()` je `null`:

```
var userName = window.prompt("What is your name?", "no name");

if (userName !== null) {
    alert("Hello " + userName);
}
```

Povratna vrednost metode `prompt()` sada se smešta unutar promenljive `userName`. Zatim se proverava da li je vrednost `userName` promenljive različita od `null`. Na taj način se utvrđuje da li je korisnik kliknuo na OK ili Cancel dugme. Ukoliko je korisnik kliknuo na dugme OK, prikazuje se prigodna poruka.

Rezime

- Web pregledači poseduju ugrađene mehanizme za kontrolu ponašanja prozora unutar koga se nalazi stranica web sajta.
- Objekt `Window` kreiraju web pregledači i unutar njega smeštaju brojne funkcionalnosti i svojstva koja se mogu koristiti za interakciju sa prozorom web pregledača ili sa samim dokumentom koji je unutar takvog prozora prikazan.
- Objekat `Window` unutar sebe objedinjuje brojne druge objekte, pri čemu svi takvi objekti nisu striktno vezani za grafičku reprezentaciju prozora i HTML dokumenta.
- Skup određenih funkcionalnosti koje se nalaze unutar `Window` objekta, a namenjene su interakciji sa samim web pregledačem, odnosno prozorom unutar koga se prikazuje HTML dokument, veoma često se kolektivno naziva *Browser Object Model*.
- `Window` objektu može se pristupiti korišćenjem promenljive `window`.
- `Window` objekat je globalni objekat web pregledača, što znači da se njegovim svojstvima i metodama može pristupati bez navođenja promenljive `window`.
- Unutar `Window` objekta nalazi se objekat `Screen`, koji omogućava pristup osnovnim informacijama o displeju korisničkog uređaja.
- Za dobijanje informacija o URL adresi na kojoj se nalazi HTML dokument koristi se objekat `Location`, koji je sastavni deo globalnog objekta `Window`.
- `Window` objekat poseduje i tri specijalne metode koje omogućavaju kreiranje modalnih prozora različitih namena: `alert()`, `confirm()` i `prompt()`.