Osnove jQueryja

Prilikom kreiranja web sajtova moguće je koristiti različite biblioteke, softverske okvire i alate. Svi oni su namenjeni olakšavanju kreiranja web sajtova ili nekih njihovih delova. Jedna od najstarijih takvih biblioteka jeste jQuery. Ona je namenjena olakšavanju definisanja klijentske logike web sajtova, i to primarno u oblastima koje su ilustrovane u prethodnom modulu, odnosno oblastima koje podrazumevaju rukovanje pretraživačem i dokumentom. Stoga će modul pred vama biti posvećen osnovama jQuery biblioteke.

Coffee Shop web sajt

Tehnike koje će biti prikazane u lekcijama ovog modula omogućiće nam da dodatno unapredimo klijentsku logiku web sajta koji razvijamo u ovom kursu. Zapravo, biće prikazani pristupi koji će nam omogućiti da već kreiranu klijentsku logiku svog Coffee Shop sajta definišemo na alternativni način, sa mnogo manje linija JavaScript koda.

Šta je jQuery?

jQuery je jedna od najstarijih i najpopularnijih JavaScript biblioteka, koju je kreirao John Resig 2006. godine. Osnovni slogan jQuery biblioteke je: *Write less, do more (Napišite manje, uradite više)*.

Slika 10.1. jQuery logo

Iz jQuery slogana može se naslutiti i osnovna namena ove biblioteke. Naime, jQuery pojednostavljuje realizaciju mnogih aspekata klijentskog web programiranja. Neke od glavnih osobina jQuery biblioteke su:

- mogućnost lake manipulacije DOM strukturom, korišćenjem mehanizama za selektovanje;
- elegantan način za definisanje JavaScript logike koja će se izvršiti prilikom pojave nekog događaja;
- uprošćen, nezavistan način za upućivanje HTTP zahteva;
- mnoštvo ugrađenih animacija;
- malo zauzeće resursa – jQuery biblioteka zauzima 30KB;
- ujednačavanje razlika koje postoje između različitih web pregledača;
- podrška za sve moderne web pregledače.

Struktura jQuery biblioteke

jQuery je biblioteka koja pripada nešto široj porodici srodnih biblioteka, koje je kreirao isti razvojni tim. Tako se jQuery porodica sastoji iz sledećih pojedinačnih biblioteka:

- **jQuery** – originalna i najpopularnija jQuery biblioteka, sa funkcionalnostima za manipulaciju DOM-om i rukovanje događajima, efektima i AJAX-om;
- **jQuery UI** – kolekcija grafičkih komponenata, animacija i tema koje se mogu koristiti prilikom kreiranja korisničkih okruženja web sajtova;
- **jQuery Mobile** – softverski okvir za kreiranje *responsive* web sajtova, sa posebnim akcentom na pametne mobilne uređaje;
- **Sizzle** – JavaScript biblioteka unutar koje su objedinjene funkcionalnosti za lako selektovanje elemenata koji se nalaze u DOM strukturi;
- **QUnit** – JavaScript biblioteka za kreiranje Unit testova.

U ovom kursu bavićemo se isključivo originalnom, izvornom jQuery bibliotekom. Ona je logički podeljena u nekoliko modula:

- **jQuery Core** – sadrži osnovne funkcionalnosti za manipulaciju DOM-om, rukovanje događajima i naprednu pretragu elemenata;
- **jQuery AJAX** – sadrži skup funkcionalnosti za lako upućivanje i obradu HTTP zahteva;
- **jQuery Effects** – skup funkcionalnosti za oplemenjivanje korisničkog okruženja jednostavnim efektima i animacijama.

Implementacija jQuery biblioteke u stranicu

Preduslov za korišćenje jQuery biblioteke jeste uključivanje jednog JavaScript fajla unutar HTML dokumenta. Prilikom uključivanja JavaScript fajla, potrebno je odlučiti se za jednu od sledeće dve verzije:

- regular build ili
- slim build.

Regular build verzija podrazumeva kompletan skup jQuery funkcionalnosti, koji pored jQuery Core modula uključuje i jQuery AJAX i jQuery Effects module. Slim build verzija podrazumeva samo jQuery Core modul.

Za koju god od ovih verzija da se odlučimo, pred nama je još jedan odabir, koji podrazumeva dve verzije:

- produkcionu verziju i
- razvojnu verziju.

Produkciona verzija se preporučuje za korišćenje na sajtovima koji su objavljeni i dostupni korisnicima. Produkciona verzija jQuery biblioteke je kompresovana i umanjena (*minified*).

Razvojna verzija jQuery biblioteke namenjena je testiranju i razvoju, te zbog toga nije

kompresovana niti umanjena, što znači da je njen kod lako čitljiv.

Razvojnu verziju možete koristiti samo ukoliko se zainteresovani za izvorni kod jQueryja, odnosno ukoliko želite da pogledate kako je neka funkcionalnost rešena unutar ove biblioteke. U svim ostalim slučajevima se preporučuje korišćenje produkcione verzije. Mi ćemo u nastavku ovog kursa koristiti isključivo produkcionu verziju.

Nakon donošenja odluke o varijanti jQuery biblioteke koja će biti korišćena, potrebno je doneti još jednu odliku – da li jQuery smestiti na isto mesto gde i web sajt ili koristiti javno dostupne CDN sisteme. Drugim rečima, za implementiranje jQuery biblioteke u stranicu postoje dva načina:

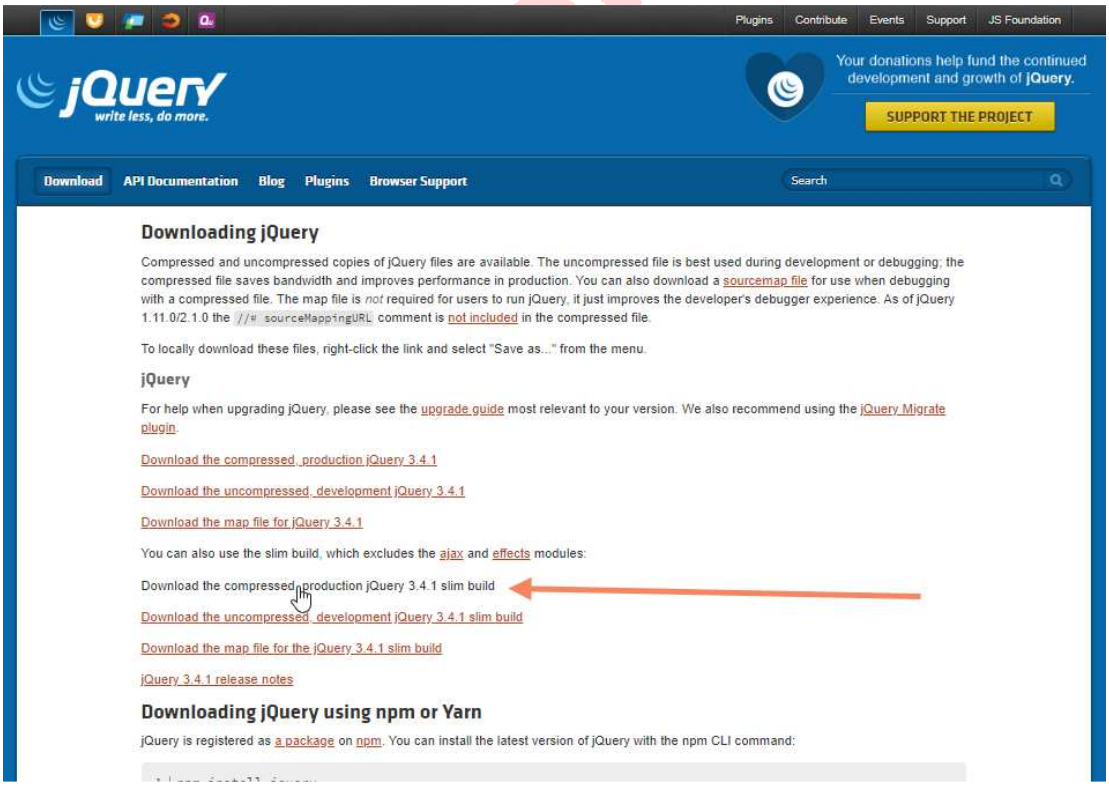
- preuzimanje fajla biblioteke sa sajta jQuery.com i njegovo uključivanje u stranicu sa lokalnog servera i
- uključivanje jQuery biblioteke hostovane na CDN serverima.

Preuzimanje jQuery biblioteke

Preuzimanje jQuery biblioteke na sopstveni kompjuter može se obaviti sa download sekcije zvaničnog web sajta:

https://jquery.com/download/

U listi različitih opcija koje se nude na zvaničnom jQuery sajtu potrebno je odabrati kompresovanu, produkcionu verziju jQuery biblioteke (slika 10.2).



Slika 10.2. Preuzimanje jQuery biblioteke

Kako bi se `.js` fajl koji predstavlja jQuery sačuvao na lokalnom kompjuteru, potrebno je izvršiti desni klik na link sa slike 10.2. i odabrati opciju *Save link as...* Na taj način će fajl koji predstavlja jQuery biblioteku biti preuzet na vaš kompjuter.

Nakon preuzimanja fajla koji predstavlja jQuery biblioteku, bilo da je reč o produkcionoj ili razvojnoj verziji, preuzeti fajl je potrebno implementirati na stranicama koje će takve funkcionalnosti koristiti. Implementiranje podrazumeva korišćenje `<script>` tagova, baš kao i prilikom uključivanja eksternih fajlova sa JavaScript kodom:

```
<script src="js/jquery-3.4.1.min.js"></script>
```

Prikazanom linijom se vrši uključivanje jQuery biblioteke koja se nalazi unutar zasebnog foldera sa nazivom `js`. Liniju za uključivanje je najbolje smestiti na sam kraj `body` elementa, ali pre bilo kog JavaScript koda koji će koristiti jQuery funkcionalnosti.

Korišćenje jQuery biblioteke sa CDN-a

Drugi način za uključivanje jQuery biblioteke u stranicu jeste korišćenje eksternih CDN servera. Naime, nekoliko velikih korporacija hostuje jQuery biblioteku na svojim serverima, odakle se ona slobodno može koristiti za integraciju unutar HTML dokumenata.

Među kompanijama koje hostuju jQuery biblioteku na svojim serverima su Google i Microsoft. Ukoliko se koristi Google CDN, uključivanje jQuery biblioteke u stranicu izgleda ovako:

```
<script
src="https://ajax.googleapis.com/ajax/libs/jquery/3.4.1/jquery.min.js"><
/script>
```

Uključivanje jQuery biblioteke korišćenjem Microsoft CDN-a:

```
<script src="https://ajax.aspnetcdn.com/ajax/jQuery/jquery-
3.4.1.min.js"></script>
```

Korišćenje CDN-a za uključivanje jQuery biblioteke poseduje određene prednosti. Biblioteke uključene na ovaj način keširaju se na klijentskim računarima samo jednom, i to kada korisnik prvi put poseti neki sajt koji koristi ovako uključenu jQuery biblioteku. Pošto je takvih sajtova zaista puno, velike su šanse da klijent već poseduje keširanu verziju biblioteke, te na taj način neće morati da je ponovo preuzima, što ubrzava učitavanje stranice. Takođe, CDN je zapravo mreža servera. Ovo praktično znači da će klijent biti opslužen sa servera koji je najbliži lokaciji klijenta, što će dodatno ubrzati učitavanje.

Referentni primer

Kako biste na najbolji način razumeli svrhu jQueryja, odmah na početku će biti prikazana dva različita načina za obavljanje jednog istog posla. Prvi način će podrazumevati korišćenje pristupa koji su već prikazani u prethodnim modulima ovoga kurs (čist JavaScript i DOM API), dok će se drugi način zasnivati na upotrebi jQueryja. HTML dokument koji će poslužiti za realizaciju primera izgleda ovako:

```
<!DOCTYPE html>
<html lang="en">

    <head>
        <meta charset="UTF-8">
        <meta name="viewport" content="width=device-width, initial-
scale=1.0">
        <title>jQuery Introduction</title>
        <style>
            input[type=text],
            input[type=email],
            select,
            textarea {
                width: 100%;
            }

            form>div {
                margin: 16px 0 16px 0px;
            }

            form>div>label {
                font-weight: bold;
            }

            form {
                max-width: 600px;
                margin: 0 auto;
            }
        </style>
    </head>

    <body>

        <form id="subscribe-form">
            <div>
                <label for="name">Name: </label>
                <input type="text" id="name" name="name">
            </div>

            <div>
                <label for="email">Email: </label>
                <input type="email" id="email" name="email">
            </div>

            <button type="submit">Subscribe</button>
        </form>

        <script>



        </script>
    </body>

</html>
```

Unutar prikazanog HTML dokumenta nalazi se jedna forma sa dva `input` elementa tipa `text` za unos imena i email adrese. Pored ova dva `input` elementa unutar forme se nalazi i jedan `button` element za prosleđivanje podataka forme. Mi podatke ove forme nećemo prosleđivati serveru, već ćemo ih obraditi lokalno korišćenjem JavaScript jezika. U dokumentu možete videti i prazan `script` element unutar koga ćemo pisati JavaScript kod:

```
let subscribeForm = document.getElementById("subscribe-
form");

subscribeForm.addEventListener("submit", function (e) {

    e.preventDefault();

    let nameInput = document.getElementById("name");
    let emailInput = document.getElementById("email");

    alert("Name: " + nameInput.value + "\n" +
        "Email: " + emailInput.value);

});
```

Za pisanje prikazanog JavaScript koda iskorišćeni su pristupi koji su obrađeni u jednom od prethodnih modula ovog kursa, u kome je bilo reči o objektnim modelima pretraživača i dokumenta i rukovanju događajima. U kodu se prvo dolazi do objektne reference `form` elementa, korišćenjem metode `getElementById()`. Zatim se obavlja registrovanje logike koja će se izvršiti prilikom prosleđivanja forme (forma se prosleđuje klikom na `button` element sa atributom `type` i vrednošću `submit`).

Unutar anonimne funkcije koja će se aktivirati prilikom prosleđivanja forme, prvo je obavljeno otkazivanje podrazumevanog ponašanja događaja. Naime, prosleđivanje forme je događaj koji se, između ostalog, automatski obrađuje od strane web pregledača, tako što se podaci iz forme upućuju na adresu definisanu `action` atributom. Pošto mi ne želimo da prosleđujemo podatke izvan okvira naše lokalne stranice, `action` atribut je izostavljen, a unutar funkcije za obradu događaja je definisan poziv metode `preventDefault()` za otkazivanje podrazumevanog ponašanja događaja.

Nakon otkazivanja podrazumevanog ponašanja događaja, unutar funkcije za obradu događaja napisan je kod kojim se dolazi do objekata koji predstavljaju *name* i *email* `input` elemente. Takvi objekti se koriste za čitanje vrednosti koje je korisnik uneo u `input` polja, za šta se koriste svojstva `value` ovih objekata. Na kraju, vrednosti koje je korisnik uneo u `input` polja prikazuju se unutar jednog modalnog prozora.

Kako biste uvideli prednosti koje donosi jQuery, sada će identičan posao biti obavljen korišćenjem jQueryja:

```
<script src="js/jquery-3.4.1.min.js"></script>
<script>
    $("#subscribe-form").submit(function (e) {
        e.preventDefault();
        alert("Name: " + $("#name").val() + "\n" +
            "Email: " + $("#email").val());
    });

</script>
```

Prvo je potrebno da primetite da je pre `script` elementa sa kodom koji smo samostalno definisali naveden i `script` element kojim se obavlja uključivanje jQuery biblioteke u HTML dokument. Veoma je bitno da se uključivanje jQueryja obavi pre koda koji će koristiti funkcionalnosti jQuery biblioteke. U protivnom, kod sa jQuery logikom neće raditi.

Unutar drugog `script` elementa nalazi se kod koji smo samostalno napisali, odnosno kod u kome se koriste funkcionalnosti jQuery biblioteke. Veoma je lako uvideti da sada postoji mnogo manje koda koji smo samostalno definisali. Eto prvog dokaza da je jQuery slogan *write less, do more* istinit. Ukoliko funkcionisanje ovakvog koda testirate u nekom web pregledaču, videćete da on proizvodi efekat identičan efektu koji proizvodi kod koji je napisan čistim JavaScriptom.

Za realizaciju prikazanog primera iskorišćeno je nekoliko osnovnih funkcionalnosti jQuery biblioteke. Kako biste bili u mogućnosti da razumete kod koji je napisan, neophodno je da se upoznamo sa takvim osnovnim osobinama jQuery biblioteke. Stoga će u nastavku ovog modula biti reči o sledećim jQuery pojmovima:

- selektori;
- događaji;
- funkcije za rukovanje strukturom, stilizacijom i sadržajem;
- funkcije za pretragu.

jQuery selektori

Kako bi se programabilno, korišćenjem JavaScript koda rukovalo HTML elementima, njih je prethodno potrebno pronaći unutar objektne reprezentacije HTML dokumenta koju web pregledači kreiraju za nas. To je u prethodnim lekcijama postizano korišćenjem funkcionalnosti koje su JavaScript kodu izložili sami web pregledači.

Biblioteka jQuery značajno pojednostavljuje proces pronalaženja elemenata unutar DOM strukture, korišćenjem takozvanih selektora, koji oponašaju CSS selektore.

jQuery selektori započinju dolar (`$`) karakterom, nakon koga slede zagrade: **`$()`**. Unutar zagrada navodi se selektor koji ukazuje na jedan ili više elemenata koji će biti selektovani. Na ovaj način se mogu kreirati tri osnovne grupe selektora, prikazane tabelom 10.1.

Kriterijum selekcije	Primer	Opis
tip elementa	`$("h1")`	selektuje sve elemente određenog tipa
vrednost id atributa	`$("#menu")`	selektuje element iz DOM strukture na osnovu vrednosti `id` atributa
naziv klase	`$(".some-class")`	selektuje sve elemente koji poseduju definisanu klasu

Tabela 10.1. jQuery selektori

U prvom jQuery primeru, koji je prikazan nešto ranije, kreirano je nekoliko jQuery selektora. Na primer, jedan od njih je:

```
$("#subscribe-form")
```

Ovakva jQuery konstrukcija ekvivalenta je sledećem JavaScript kodu:

```
document.getElementById("subscribe-form")
```

U prikazanom primeru, definisana su još dva jQuery selektora:

- `$("#name")` isto je što i `document.getElementById("name")`
- `$("#email")` isto je što i `document.getElementById("email")`

Sva tri jQuery selektora koja su upotrebljena u prvom primeru selektuju elemente na osnovu vrednosti `id` atributa. Upravo zbog toga se unutar zagrada navodi vrednost `id` atributa koja započinje hashtag karakterom (#). Pored takvog pristupa, jQuery omogućava da se elementi selektuju i na osnovnu ostalih osobina (klasa i tipova elemenata). Drugim rečima, jQuery sintaksa za selektovanje oponaša izvornu CSS sintaksu za kreiranje selektora.

jQuery funkcije za selektovanje elemenata

Odmah na početku je potrebno razumeti da jQuery nije nikakav programski jezik, već samo JavaScript biblioteka. Web pregledači razumeju isključivo JavaScript jezik, što znači da je i kod koji koristi jQuery ništa drugo do JavaScript kod, za čije formiranje se moraju poštovati sva sintaksna pravila JS jezika.

jQuery selektori možda izgledaju kao neki leksički element jezika koji do sada niste viđali, ali to svakako nisu. Ukoliko malo bolje pogledate strukturu jQuery selektora, moći ćete da prepoznate jedan od osnovnih sastojaka JavaScript jezika – funkciju:

```
$()
```

jQuery selektori su klasične JavaScript funkcije sa nazivom `$` i parametrom tipa string koji se odnosi na jedan ili više elemenata koje je potrebno selektovati.

Pored funkcije za selektovanje elemenata koja je prikazana u prethodnim primerima, jQuery poseduje još jednu funkciju identične namene:

```
jQuery()
```

Obe funkcije su potpuno ravnopravne i njima se postiže identičan efekat.

Pitanje

Koji se karakter koristi kao prefiks unutar jQuery selektora ukoliko je potrebno selektovati elemente po nazivu klase?

- **.**
- :
- #
- $

Objašnjenje:

Karakter koji se koristi kao prefiks unutar jQuery selektora ukoliko je potrebno selektovati elemente po nazivu klase je karakter tačka (.).

jQuery Object

U dosadašnjem toku ove lekcije predstavljeni su jQuery selektori. Mogli ste videti na koji način oni omogućavaju da se dođe do nekog od objekata iz objektne strukture elemenata, odnosno unutar DOM-a. Ipak, ono što do sada nije rečeno jeste to da se selektovanjem DOM elemenata korišćenjem jQuery selektora dobija jedna posebna vrsta objekata – jQuery objekat.

```
let nameInput = $("#name");
```

U prikazanoj naredbi iskorišćen je jQuery selektor iz prethodnog primera. Ipak, ovoga puta je povratna vrednost selektora spakovana unutar jedne JavaScript promenljive. Može se reći da je vrednost promenljive `nameInput` jedan jQuery objekat.

jQuery objekti unutar sebe objedinjuju sledeće:

- jedan ili više izvornih DOM objekata koji predstavljaju selektovane elemente;
- mnoštvo dodatnih svojstava i metoda koje je moguće koristiti nad takvim elementima.

Postojanje jQuery objekta je razlog zbog koga je moguće napisati nešto poput:

```
nameInput.length //1
```

Svi jQuery objekti poseduju svojstvo `length`, kojim se može dobiti informacija o broju selektovanih elemenata. U primeru iz ove lekcije, svojstvo `length` će imati vrednost `1`, zato što se prikazanim jQuery selektorom selektuje samo jedan element. Ipak, ukoliko se definiše nešto drugačiji selektor, situacija će biti drugačija:

```
let inputs = $("input");
```

Sada se selektovanje obavlja korišćenjem tipskog selektora, pa se tako selektuju svi `input` elementi na stranici. Unutar promenljive `inputs` se ponovo smešta jedan jQuery objekat. Ipak, ovoga puta je vrednost njegovog svojstva `length` drugačija:

```
inputs.length //2
```

Sada je vrednost svojstva `length` jednaka 2, s obzirom na to da je prikazanim selektorom obavljeno selektovanje dva elementa.

jQuery objekti imaju strukturu koja liči na nizove. Na početak jQuery objekata se smeštaju reference na selektovane elemente. Upravo zbog toga je moguće napisati nešto ovako:

```
let inputs = $("input");
let domObject = inputs[0];
```

Korišćenjem indeksa 0, sada je obavljen pristup prvom elementu jQuery objekta iz primera. Na taj način je unutar promenljive `domObject` smešten izvorni DOM objekat, kojim je prvi `input` element predstavljen u objektnoj strukturi dokumenta. Bitno je razumeti da takav objekat nije jQuery objekat, već izvorni DOM objekat.

Ukoliko je ovo isto potrebno obaviti, ali na kraju dobiti jQuery objekat koji reprezentuje samo jedan element iz skupa selektovanih elemenata, moguće je iskoristiti jQuery funkciju `eq()`:

```
let inputs = $("input");
let jQueryObject = inputs.eq(0);
```

Sada je unutar promenljive `jQueryObject` smešten jedan jQuery objekat koji predstavlja prvi selektovan element iz skupa selektovanih elemenata.

jQuery objekti pored referenci na izvorne DOM objekte sadrže i više od 140 metoda koje je moguće koristiti za manipulaciju DOM strukturom. Tako su jQuery objekti osnovni mehanizam koji omogućava realizaciju svih pristupa koji će biti obrađeni u lekcijama koje slede.

Rezime

- jQuery je jedna od najstarijih JavaScript biblioteka, koju je kreirao John Resig 2006. godine.
- jQuery pojednostavljuje realizaciju praktično svakog aspekta JavaScript programiranja.
- Biblioteka jQuery značajno pojednostavljuje proces selekcije elemenata, korišćenjem takozvanih selektora, koji oponašaju CSS selektore.
- Obrada događaja je još jedan aspekt programiranja koji je znatno olakšan korišćenjem jQuery biblioteke.
- Biblioteka jQuery značajno pojednostavljuje i proces upućivanja asinhronih zahteva serveru.
- jQuery selektori započinju dolar ($) karakterom, nakon koga slede zagrade: `$()`.
- jQuery rukuje specijalnim jQuery objektima.