Cookies

Pored različitih mehanizama za slanje i dobijanje podataka od servera, prilikom razvoja web aplikacija veoma korisni su i različiti pristupi koji omogućavaju čuvanje podataka na klijentu. Znate da je klijent zapravo web pregledač, pa će tako lekcije ovog modula biti posvećene različitim pristupima za čuvanje podataka unutar web pregledača. Web pregledači obezbeđuju različite mehanizme za čuvanje podataka na klijentu. Jedan od najjednostavnijih zasniva se na upotrebi kolačića, koji će biti tema ove lekcije.

Šta su kolačići?

Kolačići (*cookies*) su kratki tekstualni podaci koji se smeštaju direktno unutar web pregledača. Reč je o pojmu koji je definisan samim HTTP protokolom, odnosno, kolačići su standardizovani RFC 6265 specifikacijom IETF organizacije.

Pošto su deo HTTP protokola, kolačići se mogu razmenjivati između servera i klijenata korišćenjem specifičnih zaglavlja. HTTP server klijentu isporučuje kolačić definisanjem `Set-Cookie` zaglavlja. Klijenti imaju mogućnost da serveru vrate kolačić definisanjem `Cookie` zaglavlja (slika 11.1).

Slika 11.1. Razmena kolačića između servera i klijenta

Koja je svrha kolačića?

Slika 11.1. ilustruje osnovni princip koji obezbeđuje razmenu kolačića između servera i klijenta. Ipak, kao logično se nameće pitanje: *Koja je svrha ovakve komunikacije, koju prati razmena kolačića?*

Kolačići imaju tri osnovna mesta primene:

- rukovanje sesijama;
- personalizacija;
- praćenje.

Korišćenje kolačića za rukovanje sesijama

Već ste mogli da pročitate da je HTTP *stateless* protokol, što praktično znači da HTTP server svaki zahtev interpretira kao potpuno izolovanu celinu. Ta osobina značajno olakšava implementaciju HTTP servera i pozitivno utiče na performanse u situacijama u kojima HTTP server opslužuje veoma veliki broj zahteva tokom kratkog vremenskog perioda, zato što server ne mora da vodi bilo kakvu evidenciju o istoriji komunikacije. Ipak, zbog ovakve osobine HTTP protokola, može se reći da HTTP serveri ne znaju gotovo ništa o klijentima koji upućuju zahteve, niti mogu da smeste više klijentskih zahteva u isti kontekst. Drugim rečima, HTTP ne nalaže čuvanje bilo kakvih podataka o jednoj sesiji.

HTTP sesija predstavlja sekvencu zahteva i odgovora koje klijenti i serveri međusobno razmenjuju. S obzirom na to da HTTP ne nalaže čuvanje podataka o takvoj interakciji između servera i klijenata, veoma česta praksa prilikom razvoja web aplikacije jeste korišćenje kolačića kako bi se očuvalo stanje sesije.

Kolačići omogućavaju serveru da klijentu pošalje određenu identifikacionu vrednost. Na slici 11.1. ona je prikazana kao ključ `session-id` sa vrednošću `12345`. U nastavku komunikacije, takvu vrednost klijent pridružuje svakom HTTP zahtevu koji upućuje serveru. Na taj način server može da identifikuje klijenta koji upućuje zahtev i da veći broj zahteva koje uputi isti klijent stavi u zajednički kontekst.

Korišćenje kolačića za personalizaciju i praćenje

Kolačići se veoma često koriste i kako bi se na klijentu sačuvala podešavanja kojima se personalizuje web sajt koji korisnik pregleda. Na primer, jednom postavljeno jezičko podešavanje može se sačuvati unutar kolačića, a pri narednom otvaranju istog sajta, na osnovu takvog kolačića web pregledač i klijentska logika će znati koji je jezik najprikladniji za prikaz tekstualnih podataka.

Kolačići se često koriste i kako bi se unutar njih smestile informacije o ponašanju korisnika na web sajtu i njegovim akcijama, kako bi se na osnovu takvih informacija dobila ideja o njegovim interesovanjima. Tako se, na primer, kolačići često koriste da prilagode prikaz reklama na sajtovima interesovanju korisnika.

Kako se kreiraju kolačići?

Postoje dva osnovna načina za kreiranje kolačića:

- na serveru, pri čemu HTTP server ili pozadinska logika kreira kolačić koji prosleđuje klijentu zajedno sa serverskim odgovorom, baš kao što je to prikazano na slici 11.1;
- na klijentu, što podrazumeva kreiranje kolačića od strane JavaScript programskog jezika.

S obzirom na to da je predmet ovog kursa frontend programiranje, u nastavku se nećemo baviti načinima za kreiranje kolačića na serveru. Ipak, sa osnovnom strukturom kolačića upoznaćemo se upravo na primeru kolačića koji se dobijaju od HTTP servera.

Anatomija kolačića

Kolačići se sastoje iz sledećih komponenata:

- naziv;
- vrednost;
- direktive.

Naziv i vrednost su dve osnovne komponente koje čine jedan kolačić, tako da je moguće kreirati kolačić i bez direktiva.

Direktive se koriste da predstave neke dodatne osobine kolačića, kao što su njegovo vreme važenja, domen na koji se odnosi, kao i način razmene i pristupa na klijentu.

Uzimajući u obzir sve ovo što je rečeno, osnovna struktura HTTP kolačića može da izgleda kao na slici 11.2.

Slika 11.2. Anatomija HTTP kolačića

Sa slike 11.2. se može videti da se naziv i vrednost međusobno razdvajaju karakterom jednako, a od potencijalnih direktiva karakterom tačka sa zapetom. Takođe, karakterom tačka sa zapetom se razdvaja i više susednih direktiva.

Primer dela jednog HTTP odgovora sa jednostavnim kolačićem može da izgleda ovako:

```
HTTP/1.1 200 OK
Content-type: text/html
Set-Cookie: id=b4sYq
```

Ovakav kolačić poseduje samo naziv i vrednost. Naziv je `id`, a vrednost `b4sYq`.

Kada se kolačić iz primera isporuči klijentu, web pregledač će ga sačuvati i nakon toga uključivati u zaglavlje svakog zahteva koji bude upućivao serveru sa koga je ovaj kolačić došao.

Vreme važenja kolačića

Upravo prikazani kolačić ne poseduje nijednu direktivu koja bi dodatno odredila period njegovog važenja. Stoga će upravo prikazani kolačić biti obrisan čim se zatvori web pregledač ili tab unutar koga se prikazuje sajt kojem kolačić pripada. Ukoliko je potrebno samostalno definisati period važenja kolačića, moguće je koristiti dve direktive:

- **Expires** – definiše datum i vreme kada će kolačić biti obrisan;
- **Max-Age** – definiše period u sekundama, nakon čijeg isteka će kolačić biti obrisan.

Primer zaglavlja kojim se definiše kolačić sa definisanim datumom i vremenom brisanja može da izgleda ovako:

```
Set-Cookie: id=b4sYq; Expires=Wed, 23 Jan 2020 03:22:00 GMT;
```

Definisanje vremena važenja kolačića u sekundama se može obaviti ovako:

```
Set-Cookie: id=b4sYq; Max-Age=60;
```

Stare verzije web pregledača, kao što su Internet Explorer 6, 7 i 8, ne podržavaju `Max-Age` direktivu, pa se kod njih ona ignoriše.

Tipovi kolačića

U zavisnosti od strukture, odnosno direktiva koje su iskorišćene prilikom njihovog kreiranja, kolačići se mogu podeliti na nekoliko osnovnih grupa:

- kolačići sesija (*session cookie*);
- permanentni kolačići;
- Secure kolačići;
- HttpOnly kolačići.

Sa kolačićima sesija i permanentnim kolačićima već smo se upoznali u dosadašnjem toku ove lekcije. Kolačići sesija su oni kod kojih vreme važenja nije eksplicitno definisano direktivama `Expires` i `Max-Age`. Stoga oni postoje dok postoji i sesija, odnosno dok korisnik ne zatvori sajt koji pregleda.

Nasuprot kolačićima sesija su permanentni kolačići, čije je vreme trajanja unapred utvrđeno, korišćenjem pristupa koju su prikazani u prethodnim redovima.

Secure kolačići su oni koji se klijentu isporučuju isključivo korišćenjem HTTPS protokola. Oni se kreiraju definisanjem **Secure** direktive:

```
Set-Cookie: id=b4sYq; Expires=Wed, 23 Jan 2020 03:22:00 GMT; Secure
```

HttpOnly kolačići su oni kolačići kojima se ne može pristupiti na klijentu korišćenjem JavaScript jezika. Upravo zbog toga se i nazivaju HttpOnly, zato što se mogu videti samo kao deo klijentskih zahteva i serverskih odgovora. HttpOnly kolačići se kreiraju definisanjem **HttpOnly** direktive:

```
Set-Cookie: id=b4sYq; Expires=Wed, 23 Jan 2020 03:22:00 GMT; HttpOnly
```

Pored nabrojanih, postoje i razne druge vrste kolačića prema njihovoj nameni i osobinama. Na primer, kolačići kojima se identifikuje klijent veoma često se nazivaju kolačići za proveru autentičnosti (*authentication cookies*).

Oblast važenja kolačića

Sledeća tema kojom ćemo se baviti odnosi se na oblast važenja jednog kolačića. Pod pojmom oblasti važenja kolačića zapravo se podrazumeva skup URL adresa na koje će kolačići biti isporučivani prilikom međusobne komunikacije servera i klijenta. Osnovni pristupi za kontrolu oblasti važenja kolačića podrazumevaju korišćenje direktiva `Domain` i `Path`.

Podrazumevano, jednom kolačiću se može pristupiti samo sa domena koji ga je kreirao. To praktično znači da, ukoliko je kolačić kreiran unutar serverskog odgovora koji je nastao kao rezultat zahteva za `mysite.com` stranicom, takvom kolačiću će moći da pristupi samo `mysite.com` stranica. Stranica `someothersite.com` neće moći da pristupi takvom kolačiću.

Takođe, kolačićima se ne može pristupiti ni sa poddomena. Stoga, ukoliko je kolačić kreirao `mysite.com`, njemu se ne može pristupiti sa poddomena `dev.mysite.com`. Kako bi se takvo podrazumevano ponašanje promenilo, moguće je koristiti **Domain** direktivu:

```
Set-Cookie: id=b4sYq; Domain=mysite.com
```

U prikazanom primeru HTTP zaglavlja kojim se kreira jedan kolačić, eksplicitno je definisana `Domen` direktiva. Bitno je znati da će na ovaj način, eksplicitnim definisanjem nekog domena, kolačići postati dostupni i svim poddomenima (npr. poddomenu `dev.mysite.com`).

Pored mogućnosti za delimičnu kontrolu domena, prilikom kreiranja kolačića moguće je definisati i deo putanje koji se mora pojaviti unutar odredišne URL adrese kako bi kolačić bio uključen u klijentski zahtev ili serverski odgovor. Tako nešto se postiže korišćenjem direktive **Path**:

```
Set-Cookie: user=John; path=/
```

Putanja koja se definiše kao vrednosti `Path` direktive mora biti relativna. U primeru je definisana korena putanja, pa će tako ovakav kolačić biti primenljiv nad svim putanjama konkretnog domena. To praktično znači da se kolačić sa definisanom `Path` direktivom primenjuje za definisanu putanju, ali i za sve putanje koje njom započinju. Tako bi se prikazani kolačić, pored korene putanje, između ostalog primenjivao i za:

```
/about-us
/our-products
/about-us/our-team
/about-us/our-tema/ceo
...
```

Pitanje

Kolačiće je moguće kreirati isključivo na serveru.

a) Tačno.
b) Netačno.

Objašnjenje:

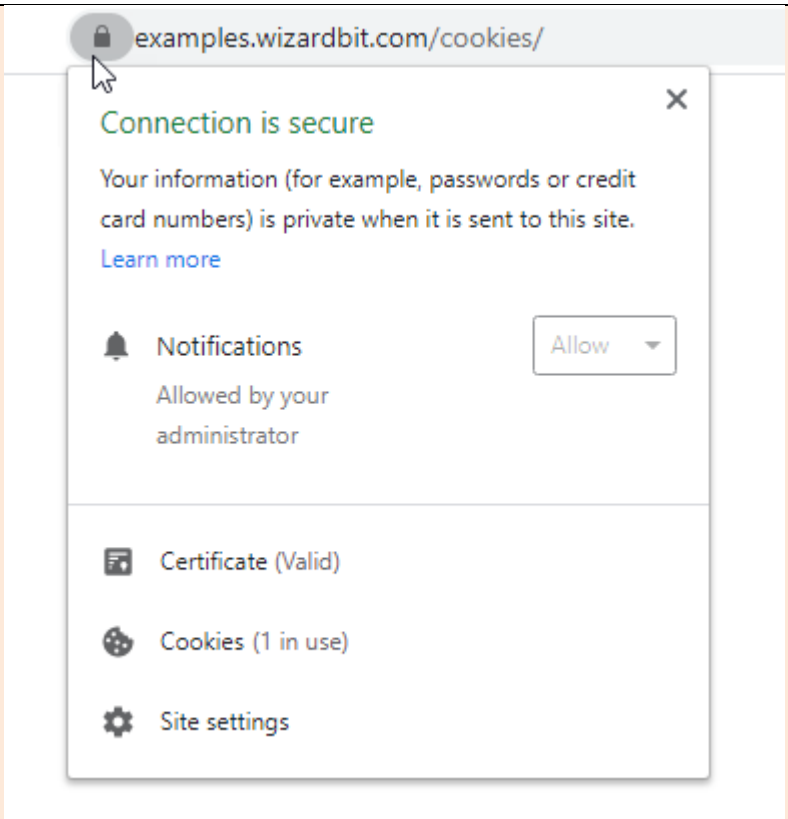
Kolačići se mogu definisati na serveru, ali i na klijentu.

Prvi primer kolačića i inspekcija njegovih osobina u pregledaču Chrome

Nakon upoznavanja osnovnih osobina kolačića, prikažimo i prvi primer komunikacije sa HTTP serverom koja je propraćena razmenom kolačića. Kolačić će biti kreiran na serveru i zajedno sa HTTP odgovorom isporučen klijentu. Sve što je potrebno da uradite jeste da posetite sledeću URL adresu:

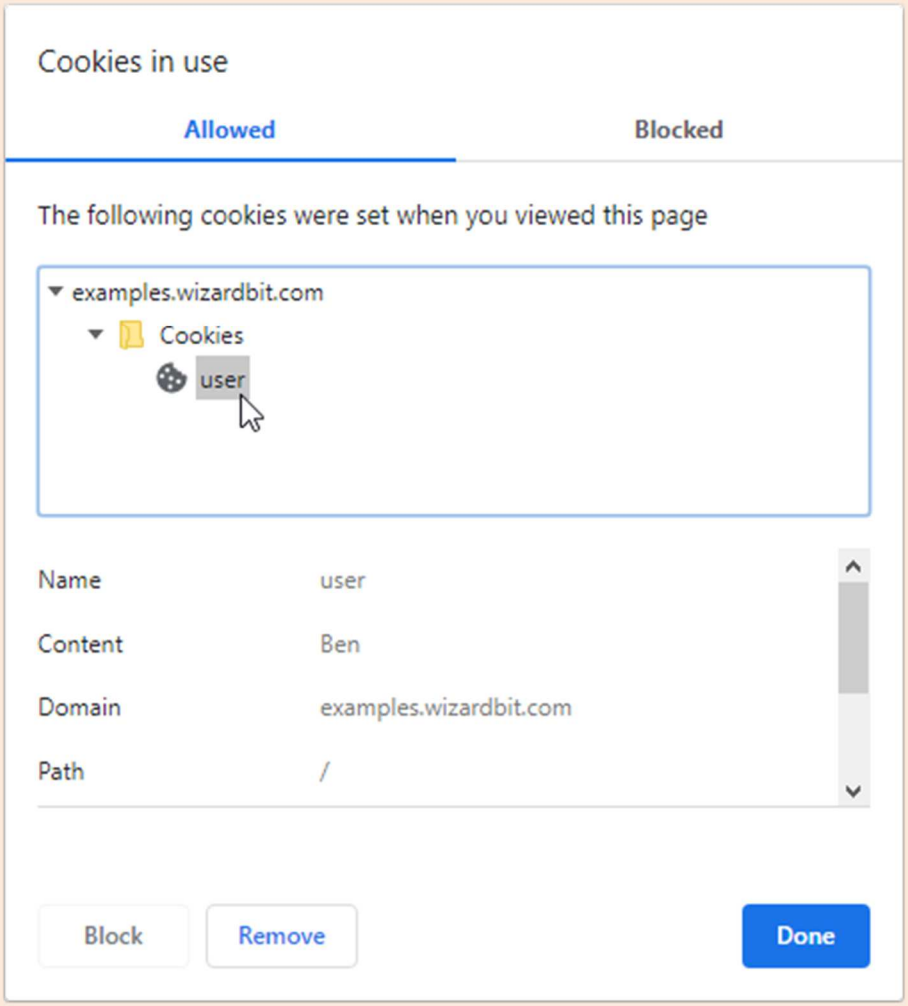
`https://examples.wizardbit.com/cookies/`

Na navedenoj URL adresi, specijalno za potrebe ove lekcije kreirana je backend logika koja HTTP odgovoru pridružuje i jedan kolačić sa nazivom `user`. Ovako kreirani kolačić na klijentu možete pregledati na nekoliko načina. U pregledaču Chrome, najjednostavniji način jeste otvaranje prozora sa informacijama o sajtu (slika 11.3).



Slika 11.3. Pregled informacija o sajtu unutar pregledača Chrome

U donjem delu prozora sa informacijama o sajtu nalazi se odeljak *Cookies*, gde je jasno napisano da je trenutno u upotrebi jedan kolačić (*1 in use*). Klikom na natpis *Cookies* otvara se prozor koji omogućava pregled informacija o svim kolačićima povezanim sa trenutnom stranicom (slika 11.4).



Slika 11.4. Pregled informacija o kolačićima unutar Chromea

Na slici 11.4. mogu se videti osnovne informacije o kolačiću koji je zajedno sa HTTP odgovorom isporučen web pregledaču.

Drugi način za pregled kolačića jeste direktna inspekcija sirovog HTTP odgovora koji se dobija od klijenta. U ovom primeru, deo HTTP odgovora koji se dobija od servera izgleda ovako:

```
HTTP/1.1 200 OK
Set-Cookie: user=Ben; expires=Sun, 31-May-2020 10:11:59 GMT; Max-
Age=2592000; path=/
```

Sada je moguće videti kompletnu strukturu kolačića koji web pregledač dobija od servera. Kao što možete videti, tu su i različite direktive o kojima je bilo reči nešto ranije u ovoj lekciji (`expires`, `Max-Age`, `path`).

Rad sa kolačićima na klijentu

Rad sa kolačićima na klijentu postiže se korišćenjem DOM (Document Object Model) API-a. Naime `Document` objekat poseduje svojstvo **cookie**, koje je moguće koristiti za čitanje postojećih i kreiranje novih kolačića.

S obzirom na to da je prethodni primer ilustrovao slanje zahteva za stranicom koja se nalazila na eksternom serveru i na domenu za čije fajlove nemamo pristup, u nastavku će rad sa kolačićima na klijentu biti ilustrovan upotrebom lokalnog HTTP servera. Instalacija lokalnog HTTP Apache (XAMPP) servera prikazana je u jednoj od prethodnih lekcija. Primer će se sastojati iz jednog dokumenta sa backend logikom napisanom PHP jezikom. Takav fajl možete da preuzmete sa sledećeg linka:

`cookies.php`

Dokument `cookies.php` potrebno je smestiti na lokalni server unutar foldera u kome će se naći i HTML dokument sa JavaScript kodom koji će biti prikazan u nastavku. Slanjem zahteva za `cookies.php` dokumentom biće kreirano nekoliko kolačića sa različitim osobinama:

```
Set-Cookie: user=Ben; expires=Sun, 31-May-2020 13:24:20 GMT; Max-Age=2592000;
path=/

Set-Cookie: color=black; expires=Sun, 31-May-2020 13:24:20 GMT; Max-
Age=2592000; path=/

Set-Cookie: code=3j2h5t24uth; expires=Sun, 31-May-2020 13:24:20 GMT; Max-
Age=2592000; path=/; HttpOnly
```

Ovakvim kolačićima ćemo u nastavku pokušati da rukujemo korišćenjem `document.cookie` svojstva.

Čitanje svih kolačića

Prvo će biti prikazan primer za čitanje svih kolačića koji su na raspolaganju našem dokumentu. Primer će biti realizovan ispisom vrednosti svojstva `document.cookie`:

```
console.log(document.cookie);
```

Na ovaj način će unutar konzole biti ispisan sledeći tekst:

```
user=Ben; color=black
```

Možete da vidite da se unutar konzole dobija tekstualni ispis kojim se svaki kolačić predstavlja kao par naziva i vrednosti, koji je od narednog takvog para odvojen karakterom tačka sa zapetom.

Bitno je da primetite da dobijamo podatke dva kolačića iako serverska skripta `cookies.php` kreira tri kolačića. Kolačić sa nazivom `code` poseduje direktivu `HttpOnly`, te stoga nije dostupan klijentskoj JavaScript logici.

Osobine document.cookie svojstva

Na osnovnu upravo prikazanog ispisa može se zaključiti da je vrednost `document.cookie` svojstva `string` tipa. Iako je na prvi pogled ova konstatacija tačna, veoma je bitno reći da je `document.cookie` jedno specijalno svojstvo sa kakvim se do sada nismo susretali.

Svojstvo `document.cookie` poseduje getter i setter metode koje su izvorno implementirane na nivou samog DOM API-a. To praktično znači da manipulacija takvim svojstvom ne predstavlja direktnu manipulaciju njegovom vrednošću. Svaki pokušaj čitanja ili postavljanja vrednosti ovog svojstva provlači se kroz unapred definisanu getter i setter logiku, koja dodatno transformiše operacije čitanja i pisanja.

Na primer, u upravo prikazanom primeru čitanja svih kolačića, mogli ste da vidite da se kao vrednost `document.cookie` svojstva dobijeni kolačići predstavljaju nazivom i vrednošću, bez bilo kakvih direktiva koje bliže određuju osobine kolačića. Naravno, takve direktive postoje, web pregledači su njih svesni, ali DOM API njih ne isporučuje kao sastavni deo vrednosti koja se dobija čitanjem `document.cookie` svojstva.

U narednim primerima kreiranja novih kolačića moći ćete da vidite još zanimljivih osobina `document.cookie` svojstva.

Kreiranje novog kolačića

Kreiranje novog kolačića na klijentu postiže se postavljanjem kolačića u tekstualnom obliku za vrednost svojstva `document.cookie`:

```
document.cookie = "lang=en";
```

Kreiranje novog kolačića još jednom potvrđuje osobinu `document.cookie` svojstva o kojoj je nešto ranije bilo reči. U normalnim okolnostima, očekivalo bi se da ovakav kod u potpunosti prepiše vrednost svojstva, ali u slučaju `document.cookie` to se ne dešava. Vrednost koja se postavlja zapravo se prosleđuje ugrađenoj setter metodi, koja na ovaj način kreira novi kolačić, dok postojeći kolačići ostaju nepromenjeni.

Ukoliko sada nakon ovakve naredbe postavimo kod za čitanje svih kolačića, dobićemo:

```
lang=en; user=Ben; color=black
```

Korišćenjem prikazanog koda, kreiran je kolačić definisanjem njegove vrednosti i naziva. Vrednosti različitih direktiva će biti podrazumevane. Tako će na ovaj način biti dobijen kolačić sa sledećim osobinama:

- naziv: lang
- vrednost: en
- domen: localhost
- putanja: /frontend/data-access/cookies
- Secure false
- HttpOnly false
- važenje – dok traje sesija pregledača

Prilikom kreiranja novih kolačića, moguće je koristiti i sve pristupe koji su ilustrovani u prvom delu ove lekcije, a koji se odnose na definisanje osobina kolačića korišćenjem različitih direktiva. Tako je moguće napisati:

```
document.cookie = "lang=en; max-age=3600";
```

Na ovaj način će biti kreiran kolačić sa dodatnom direktivom koja određuje period njegovog važenja. S obzirom na to da je iskorišćena `max-age` direktiva, na prikazani način je definisano da će kolačić postojati jedan sat (60 sekundi * 60 minuta = 3600) nakon kreiranja.

Na identičan način je moguće postaviti i ostale direktive. Jedina direktiva koja se ne može postaviti upotrebom JavaScript jezika je `HttpOnly`, s obzirom na to da je ona rezervisana samo za kolačiće koji postoje na nivou HTTP protokola.

Napomena

Svojstvo `document.cookie` omogućava kreiranje samo jednog kolačića u jednom trenutku. Drugim rečima, nije moguće definisati dva kolačića odjednom, već se kreiranje svakog kolačića mora definisati u zasebnoj naredbi.

Ažuriranje postojećeg kolačića

Svojstvo `document.cookie` omogućava i ažuriranje osobina već kreiranih kolačića, i to na isti način na koji se obavlja njihovo kreiranje. Kada se na već prikazani način pokuša definisanje kolačića koji je već kreiran, umesto kreiranja novog kolačića biće obavljeno ažuriranje postojećeg:

```
document.cookie = "lang=sr; max-age=7200";
```

Na ovaj način će biti ažuriran kolačić sa nazivom `lang`. Njegova dosadašnja vrednost (`en`) biće promenjena na `sr`, a period važenja na 7200 (dva sata).

Resetovanje prethodno kreiranog kolačića

Svojstvo `document.cookie` ne omogućava izvorni način na koji bi se neki kolačić obrisao. Ipak, korišćenjem mogućnosti koje su nam na raspolaganju, moguće je obaviti resetovanje kolačića, što će na kraju rezultovati njegovim brisanjem od strane web pregledača. Nama je na raspolaganju mogućnost izmene vrednosti i osobina kolačića, pa je moguće napisati ovako nešto:

```
document.cookie = "lang= ; expires = Thu, 01 Jan 1970 00:00:00 GMT";
```

Na ovaj način uklanja se vrednost kolačića sa nazivom `lang`. Pored toga, što je mnogo značajnije, isticanje kolačića se postavlja na vreme u prošlosti. Na ovaj način će web pregledač prilikom narednog osvežavanja stranice ukloniti kolačić sa imenom `lang`.

Provera postojanja kolačića po nazivu

U prethodnim redovima ste mogli da vidite da rad sa kolačićima na klijentu podrazumeva rukovanje svojstvom `document.cookie`. Pored pristupa za kreiranje i ažuriranje postojećih kolačića, svojstvo `document.cookie` ne obezbeđuje nikakve dodatne mehanizme koji bi olakšali neke druge aspekte rada sa kolačićima na klijentu. Stoga, bilo koji napredniji zahvat nad kolačićima na klijentu zahteva ručno pisanje logike koja će obraditi vrednost koju dobijamo čitanjem `document.cookie` svojstva. Na primer, ukoliko želimo da proverimo postojanje nekog konkretnog kolačića na osnovu njegovog naziva, takvu logiku moramo samostalno da definišemo:

```
function cookieExists(name) {
      if (document.cookie.split(';').some((item) =>
item.trim().startsWith(name + '='))) {
            return true;
      }

      return false;
}
```

Prikazani kod ilustruje funkciju `cookieExists()` sa logikom koja proverava da li kolačić sa prosleđenim nazivom postoji. Provera se postiže parsiranjem vrednosti svojstva `document.cookie`. Njegova vrednost se prvo deli po karakteru tačka sa zapetom, a zatim se vrši provera početne vrednosti svakog člana tako odbijenog niza. Ukoliko se pronađe podudaranje sa prosleđenom vrednošću, funkcija `cookieExists()` emituje vrednost `true` kao signal da kolačić sa prosleđenim imenom postoji.

Rezime

- Kolačići su kratki tekstualni podaci koji se smeštaju direktno unutar web pregledača.
- Kolačići imaju tri osnovna mesta primene: rukovanje sesijama, personalizacija i praćenje.
- Kolačići se mogu kreirati na serveru, ali i na klijentu.
- Kada se kolačić isporuči klijentu, web pregledač ga čuva i nakon toga uključuje u zaglavlje svakog zahteva koji upućuje serveru sa koga je taj kolačić došao.
- Kolačići se sastoje iz naziva, vrednosti i direktiva.
- Naziv i vrednost su dve osnovne komponente koje čine jedan kolačić, tako da je moguće kreirati kolačić i bez direktiva.
- Direktive se mogu koristiti za definisanje dodatnih osobina kolačića: vreme važenja, domen, način razmene, način pristupa...
- Kolačić bez direktive koja određuje njegovo važenje briše se čim se zatvori prozor pregledača u kome je prikazana web stranica.
- Samostalno definisanje perioda važenja kolačića može se obaviti korišćenjem direktiva `Expires` i `Max-Age`.
- Kolačići sesija su kolačići kod kojih vreme važenja nije eksplicitno definisano.
- Permanentni kolačići su oni čije je vreme trajanja unapred utvrđeno.
- Secure kolačići su oni koji se klijentu isporučuju isključivo korišćenjem HTTPS protokola, upotrebom direktive `Secure`.
- HttpOnly kolačići su oni kolačići kojima se na klijentu ne može pristupiti korišćenjem JavaScript jezika; definišu se korišćenjem `HttpOnly` direktive.
- Kontrola oblasti važenja kolačića zasniva se na korišćenju direktiva `Domain` i `Path`.

- `Document` objekat poseduje svojstvo `cookie`, koje je moguće koristiti za čitanje postojećih kolačića i kreiranje novih.
- Svojstvo `document.cookie` poseduje getter i setter metode koje su izvorno implementirane na nivou samog DOM API-a, pa se one koriste da dodatno prilagode logiku upisa i isporuke vrednosti ovog svojstva.