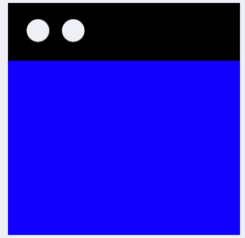
Code Academy

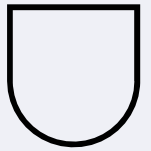


Gytis Juozenas

Duomenų saugojimas vartotojo pusėje

2022

JavaScript programavimo kalba



Šiandien išmoksite

01 Kas yra Javascript **Session Storage**

02 Kas yra Javascript **Local Storage**

Session Storage vs Local Storage

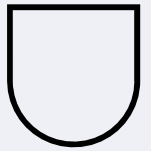
LocalStorage ir **SessionStorage** yra objektai, kurie kaip ir Slapukai gali saugoti informaciją vartotojo pusėje.
Abu objektai taip pat yra saugomi **key** ir **value** poromis duomenų tipais **string**, tik kitaip nei **slapukai (cookies)**.

Abu objektai turi tokius pačius metodus ir savybes:

- **setItem(key, value)** – store key/value pair.
- **getItem(key)** – get the value by key.
- **removeItem(key)** – remove the key with its value.
- **clear()** – delete everything.
- **key(index)** – get the key on a given position.
- **length** – the number of stored items.

SessionStorage egzistuoja tol, kol nėra uždaromas langas, naršyklė. Duomenys išsisaugo refresho metu, bet jeigu esame viename tinklapyje skirtinguose languose – bus dvi skirtingos sesijos, todėl sessionStorage yra naudojamas daug rečiau.

LocalStorage egzistuoja visada, kol neištrinamas specialiai. Duomenys išsaugomi skirtinguose languose, jei naršoma tame pačiame tinklapyje. Duomenys gali likti išsaugoti ir po operacinės sistemos restart'o.



Duomenų saugojimas ir gavimas

Išbandykime:

localStorage.setItem('test', 1);

Uždarykite langą.

Atsidarykite tą patį adresą.

console.log(localStorage.getItem('test'))

Tą patį atlikite su sessionStorage

Strings only

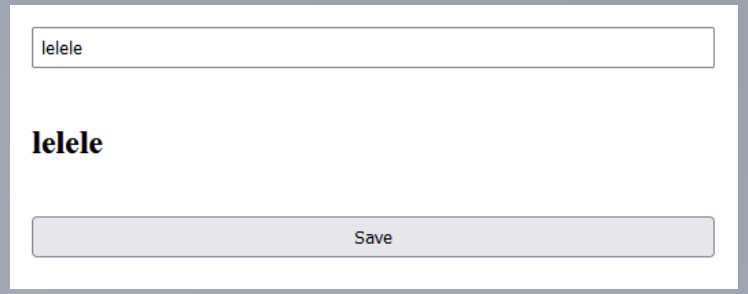
Please note that both key and value must be strings.

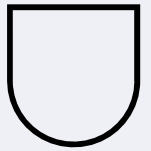
If were any other type, like a number, or an object, it gets converted to string automatically:

```
1  localStorage.user = {name: "John"};
2  alert(localStorage.user); // [object Object]
```

Practice Time

1. Sukuriame formą su vienu input type text, h2 elementą formoje ir submit mygtuką.
2. JS apsirašome keyboard event, kad atleidus klaviatūros mygtuką iškarto būtų atvaizduojamas vedamas tekstas ir nereiktų spausti Save. (keyup event).
3. Submit mygtukui reikia apsirašyti atskirą event, kad visa informacija išsisaugotų LocalStorage.





window object ir onload()

window object:

window objektas susideda iš screen ir document objektų. window objektas reprezentuoja atidarytą browser'io tab'ą.

Jei dokumente yra <iframe> tag'ų tuomet naršyklė sukuria vieną window objektą HTML dokumentui ir po vieną kiekvienam <iframe>.

Paprasčiausias pavyzdys su window objektu čia.

onload() event:

onload event suveikia tuomet kai puslapis pilnai pasikrauna. Kitaip sakant, kažkas (tai kas aprašyta) įvyks tik tuomet kai puslapis bus pilnai pakrautas. Pavyzdžių naudojant event handlerį ir ne tik galite rasti čia.

window.onload() vs document.onload():

window.onload pradės vykdyti tuomet kai visas pilnai puslapis bus pakrautas, įskaitant jo visą turinį (nuotraukos, CSS, skriptai ir t.t.).

document.onload pradės vykdyti anksčiau nei window.onload, nes jam užteks kad tik DOM būtų pakrautas, į ką neįeina joks external turinys, kaip kad nuotraukos, skriptai ir t.t.

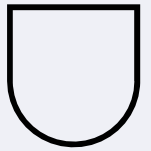
Daugiau apie jų skirtumą galite rasti čia.

Practice Time

Čia tokia dviguba užduotis, padarykite ją su localStorage iš pradžių, o po to su sessionStorage (minimaliai tereiks pakeisti):

1. Sukurkime situaciją, kuomet atėjus į puslapį atsiranda popup su prašymu sutikti su terminais ir sąlygomis (Agree to terms and conditions). Vietoj įprasto alert() naudokime confirm().
2. Jeigu vartotojas sutinka – sutikimas įdedamas į localStorage, pvz. consent: 'accepted';
3. Jeigu vartotojas nesutinka – niekas neįvyksta.
4. Uždarome langą ir atidarome iš naujo pasitikrinti.
5. Įsitikinus, kad veikia, localStorage pakeičiame sessionStorage.
6. Ir galutinis punktas tai pagalvokite koks skirtumas čia tarp localStorage ir sessionStorage.

Hints:
Nepamirškite čia panaudoti onload event‘o.



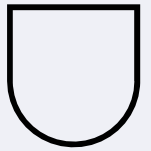
JSON

JSON – JavaScript Object Notation (more here)
Duomenų atvaizdavimo būdas naudojamas labai labai daug kur, vos ne viskas yra saugojama JSON formato failuose, kaip kad API (Application Programming Interface), configuration failai, formų užpildymai ir t.t. JSON yra patogus dar tuomet kad iš esmės beveik visos programavimo kalbos turi būdą kaip integruoti JSON, kad jis galėtų būti panaudojamas.

JSON gali laikyti šiuos duomenų tipus:
- Strings;
- Numbers (neigiami, teigiami, su ar be kablelio ir net scientific notation);
- Booleans;
- null;
- Arrays;
- Objects { “key”: “value”} {“age”: 30}.

Failo formatas: name.json
Pvz.: [

```
    {
      “name”: “Amy”,
      ”age”: 18, ”
      online”: false,

      “languages” : [“English”, “Lithuanian”, “German”],
    }
  ]
```



Kitų duomenų tipų saugojimas su JSON

O jeigu norite išsaugoti daugiau duomenų, pavyzdžiui visą objektą, ar masyvą, nes iki šiol mes viską tegalėjome saugoti string duomenų tipu?

Į pagalbą atkeliauja **JSON.stringify()** ir **JSON.parse()** metodai:
JSON.stringify() – paverčia object į string.
JSON.parse() – paverčia string į object, tačiau negali būti bet koks string, jis turi būti object tipo/formato.

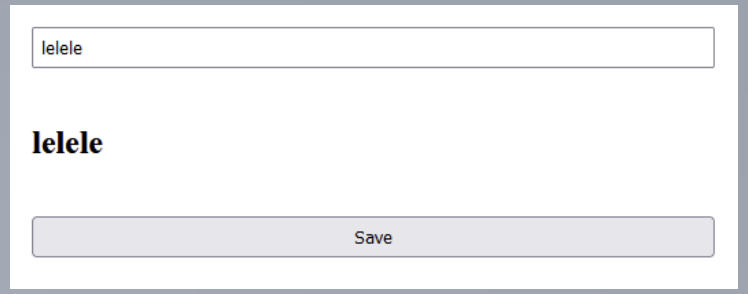
Pvz.:
```
const person = {
    name: 'Amy‘,
    age: 18, ”
    online”: false,
}

const string = JSON.stringify(person)
console.log(string, typeof string)
```

Online JSON Editor & **JSON Recap**

Practice Time

1. Sukuriame formą su vienu input type text, h2 elementą formoje ir submit mygtuką.
2. JS apsirašome keyboard event, kad atleidus klaviatūros mygtuką iškarto būtų atvaizduojamas vedamas tekstas ir nereiktų spausti Save. (keyup event).
3. Submit mygtukui reikia apsirašyti atskirą event, kad visa informacija išsisaugotų LocalStorage.

4. Pakoreguokite prieš tai darytą užduotį su input tekstu.
5. Kiekvieną kartą paspaudus save, yra pridedamas vis naujas tekstas į tekstų masyvą, kuris saugojamas localStorage. Paėmus palyginimui, prieš tai išsisaugodavo vienas key kurio value vis atsinaujindavo iš naujo įvedus kokią nors reikšmę į input.
6. Save paspaudimo metu atvaizduojamas atsitiktinis įrašas iš localStorage išsaugotų tekstų.



HINTS:
Jums prireiks tokios komandos:
JSON.parse(localStorage.getItem('inputTextName')) || [];

Practice Time

Užduoties tikslas yra kad vartotojas galėtų sukurti korteles į kurias galėtų įrašyti tekstą iš vienos kortelės pusės ir iš kitos:

1. Sukurkite formą kurioje bus galima įrašyti tekstą kuris bus atvaizduotas kortelės priekyje ir gale bei būtų du mygtukai vienas leidžiantis išsaugoti kortelę, kitas peržiūrėti korteles.
2. Paspaudus išsaugoti kortelę ji išsisaugos localStorage ir paspaudus peržiūrėti korteles atsidarys kitas HTML failas ir jame matysis kortelės priekis o po kortele bus mygtukas kuris apsuka kortelę ir parodo kas parašyta iš galo. Taip pat šalia bus mygtukas kuris leis grįžti į praeitą HTML failą ir pridėti daugiau kortelių.
3. Initially kortelių peržiūrėjimo HTML faile turėtų rodyti ale nulinę, t.y. tuščią kortelę ir kiekvieną kartą paspaudus ant kortelės turėtų pereiti į kitą kortelę ir taip rotuotis iš eilės. Jeigu yra sukurta tik viena kortelė tai iš tuščios ateis į sukurtą ir kiek bespaudinėsit niekas nekis nes liks ta vienintelė kortelė, tačiau jeigu yra daugiau nei viena kortelė tuomet eis viena po kitos.

Papildomai:

1. Delete Functionality;
2. Edit Functionality;
3. Add Images to the FlashCards;
4. Change Styling.