

JavaScript ohjelmointi

JSON & LocalStorage

MARGIT TENNOSAARIN MATERIAALISTA
LAURA JÄRVISEN MUOKKAAMA

JSON

Mikä JSON?

- JSON (*JavaScript Object Notation*) on kevyt dataformaatti, jota käytetään **tietojen siirtämiseen palvelimen ja asiakkaan välillä** tai eri ohjelmointiympäristöjen kesken.
- Vaikka JSON pohjautuu JavaScriptiin, se on **kieliriippumaton formaatti**, mikä tekee siitä laajasti käytetyn web-kehityksessä.

JSON käyttötapauksia

- API:t* ja verkkopalvelut – JSON on vakioformaatti tiedonvaihdossa asiakkaiden ja palvelimien välillä.
- Konfiguraatiotiedostot – Käytetään asetuksissa ja sovellusten konfiguraatioissa (esim. package.json Node.js:ssä).
- Tietojen tallennus – NoSQL-tietokannat kuten MongoDB käyttävät JSON-tyylistä formaattia tietojen tallentamiseen.

*API

API (*Application Programming Interface*) on ohjelmointirajapinta, joka määrittelee joukon sääntöjä ja toimintoja, joiden avulla erilaiset ohjelmistot voivat kommunikoida keskenään.

Se tarjoaa tapoja, joiden avulla sovellukset voivat käyttää toistensa toiminnallisuutta tai dataa ilman että niiden tarvitsee tuntea tai ymmärtää toistensa sisäistä rakennetta.

JSON hyödyt

- Monialustainen – Voidaan käyttää lähes minkä tahansa ohjelmointikielen kanssa.
- Ihmislueuttava – Yksinkertainen ja helppo ymmärtää.
- Kevyt – Kompaktimpi kuin XML, mikä vähentää verkon kuormitusta.

JSON syntaksi

- **Oliot:** Suljettu aaltosulkeisiin { } ja sisältävät avain-arvo-pareja. Avain lainausmerkkien sisällä
- Pilkku erottaa tiedot toisistaan
- **Taulukot:** Suljettu hakasulkeisiin [] ja sisältävät useita arvoja
- **Merkkijonot:** On suljettava lainausmerkkeihin " "
- **Numerot, totuusarvot (Boolean) ja null:**
Esitetään samalla tavalla kuin JavaScriptissä

```
{"name": "John Doe",  
  "age": 30,  
  "isStudent": false,  
  "courses": ["Math", "Science"],  
  "address": {"city": "New York",  
              "zipcode": "10001"}}
```

JS olion muuttaminen JSON dataksi

- Kun työskentelet JSONin kanssa JavaScriptissä, sinun täytyy usein muuntaa tietoa JavaScript-olioiden ja JSON-merkkijonojen välillä.
- Käytä `JSON.stringify()`

```
const person = {name: "John Doe", age: 30};  
const jsonString= JSON.stringify(person);  
console.log(jsonString); // Tulostaa: '{"name":"JohnDoe","age":30}'
```

JSON tiedon muuttaminen JS olioksi

- Palvelimelta saatu JSON data tulee muuttaa JavaScript olioksi, jotta sitä voidaan käsitellä ohjelmassa
- Käytä JSON.parse()

```
const jsonString= '{"name":"JohnDoe","age":30}';  
const personObject= JSON.parse(jsonString);  
console.log(personObject); // Tulostaa: { name: "John Doe", age: 30 }
```

Tallennus ja haku JSON-datan avulla

- JSONia käytetään usein tietojen tallentamiseen LocalStorageen tai tietojen lähettämiseen API:ssa.

```
const settings = { theme: "dark", language: "English" };  
  
// Muunna olio JSONiksi ja tallenna se  
localStorage.setItem("settings", JSON.stringify(settings));
```

Tietojen hakeminen ja jäsentäminen JSON-muodosta

- Oliot tallentaa usein taulukoihin, varsinkin tosielämän toteutuksissa

```
const storedSettings= JSON.parse(localStorage.getItem("settings"));  
console.log(storedSettings.theme); // Tulostaa: "dark"
```

JSONin rajoituksia

- JSON ei tue funktioita tai undefined-arvoja.
- JSON ei tue Date-olioita, joten ne täytyy tallentaa merkkijonoina ja muuntaa takaisin Date-olioiksi.

LocalStorage

Mikä LocalStorage?

LocalStorage on verkkoselaimen tallennusrajapinta (API), joka mahdollistaa avain-arvoparien tallentamisen selaimeen.

Toisin kuin evästeet (*cookies*), LocalStorage säilyttää tiedot, vaikka selain suljettaisiin ja avattaisiin uudelleen.

Mikä LocalStorage?

- Pysyvä tallennus – Data säilyy tallessa, kunnes se poistetaan manuaalisesti.
- Verkkotunnus*- eli domainkohtainen – Data on käytettävissä vain siitä domainista, joka sen tallensi.
- Tallennusrajoitus – Voi tallentaa noin 5MB tietoa.
- Synkroninen API – Toiminnot suoritetaan välittömästi, mikä voi vaikuttaa suoritussykyyn, jos tietoa on paljon

*Domain eli verkkotunnus

- Verkkotunnus eli domain on ihmisen luettava nimi, joka ohjaa selaimen oikeaan palvelimeen internetissä.
- DNS (Domain Name System) kääntää sen IP-osoitteeksi, jotta selain löytää palvelimen.
- Domainit rekisteröidään ylläpidon kautta, ja siihen voi liittää esimerkiksi sähköpostipalvelut, aliverkkotunnukset ja SSL-sertifikaatit.

LocalStorage perustoiminnot

- LocalStorage tarjoaa neljä pääasiallista metodia:
 - **setItem()**
 - **getItem()**
 - **removeItem()**
 - ja **clear()**

Tiedon tallennus

- Käytä **setItem()**-metodia tallentaaksesi tietoa LocalStorageen.

```
localStorage.setItem('username', 'JohnDoe');  
localStorage.setItem('theme', 'dark');
```

- Jos tallennat olioita tai taulukoita, muunna ne merkkijonoiksi käyttäen **JSON.stringify()**

```
const user = { name: 'John', age: 30 };  
localStorage.setItem('user', JSON.stringify(user));
```

Tiedon hakeminen

- Käytä **getItem()**-metodia hakeaksesi tallennettua tietoa

```
const username = localStorage.getItem('username');  
console.log(username); // Tulostus: JohnDoe
```

- Jos haet olion tai taulukon, käytä **JSON.parse()** palauttaaksesi alkuperäisen muodon.

```
const userData = JSON.parse(localStorage.getItem('user'));  
console.log(userData.name); // Tulostus: John
```

Tiedon poistaminen

- Tietyn tiedon poistamiseen käytä **removeItem()**

```
localStorage.removeItem('username');
```

- Kaiken tiedon poistamiseen käytä **clear()**

```
localStorage.clear();
```

Tallenna

```
function savePreferences() {  
  const theme = document.querySelector('#theme').value;  
  const username = document.querySelector('#username').value;  
  localStorage.setItem('theme', theme);  
  localStorage.setItem('username', username);  
  alert('Preferences saved!');  
}
```

Hae tallennetut tiedot

```
function loadPreferences() {  
  const savedTheme= localStorage.getItem('theme');  
  const savedUsername= localStorage.getItem('username');  
  if (savedTheme&& savedUsername) {  
    document.querySelector('#theme').value = savedTheme;  
    document.querySelector('#username').value = savedUsername;  
    alert(`Welcome back, ${savedUsername}!`);  
  }  
}
```

Poista tiedot

```
function clearPreferences() {  
  localStorage.removeItem('theme');  
  localStorage.removeItem('username');  
  alert('Preferences cleared!');  
}
```

Käyttötapauksia

- Käyttäjäasetusten tallentaminen (esim. teema, kieli).
- Pienen datan tallentaminen välimuistiin API-kutsujen vähentämiseksi.
- Käyttäjän edistymisen seuranta peleissä tai lomakkeissa.
- Väliaikaisen istuntodatan tallentaminen sivun latausten välillä.

LocalStorageen rajoituksia

- Rajoitettu tallennustila (~5 Mt).
- Synkroninen lukkiutuminen eli toiminnot keskeyttävät skriptin suorittamisen.
- Tietoturvariski: Tallennettu data on JavaScriptin kautta saavutettavissa, mikä altistaa XSS-hyökkäyksille.
- Same-origin-käytäntö eli data on käytettävissä vain samasta verkkotunnuksesta ja protokollasta.

Harjoitellaan

Löydät tehtäviä JSONin ja LocalStoragen harjoittelemiseen viikon 8 kansioista

<https://github.com/bc-web-ohjelmistokehitys/WP25K-JS>

Huom! Opettele itse - älä pyydä tekoälyä ratkaisemaan tehtävää puolestasi.