Spred a rest operátory

Syntax oboch týchto operátorov je rovnaká (...) no rozdielne je ich využitie.

Rest

Predstavme si, že chceme deklarovať funkciu sum s dvoma parametrami, čo by mohlo vyzerať nasledovne:

```
function sum(num1,num2){
    return num1 + num2
}
sum(1,2) //3
```

Čo ak by sme ale chceli sčítať nie 2 čísla ale n čísel. Jedným z riešení by bolo deklarovať funkciu tak aby akceptovala jeden argument typu pole čo by mohlo vyzerať nasledovne:

```
function sumArray(array){
    let sum=0;
    array.forEach(item=>sum=sum+item)
    return sum
}
sumArray([1,2,3,4]) //10
```

V JS ale poznáme aj vstavané metódy, ktoré akceptujú n argumentov (nie jeden typu pole) ako napríklad:

```
> console.log(Math.max(1,2,3,4))
  4
```

Rest operátor ponúka spôsob ako takúto funkciu deklarovať, s nasledujúcou syntaxou:

```
function sumN(...numbers){
    let sum=0;
    numbers.forEach(number=>sum=sum+number)
    return sum
}
sumN(1,2,3,4) //4
```

Všimnime si rozdiel pri volaní funkcie sumArray a sumN, zatiaľ čo pri volaní funkcie sumArray sme odovzdávali 1 argument (pole obsahujúce čísla, ktoré sme chceli sčítať) pri volaní funkcie sumN sme odovzdali 4 argumenty (obsahujúce čísla, ktoré sme chceli sčítať). Rest operátor nám teda umožňuje vytvárať funkcie akceptujúce n argumentov.

Syntax rest operátora - operátor možno využiť len pred posledným parametrom funkcie!

```
function functionN (parameter1, parameter2, ...rest){
    //telo funkcie
}
functionN(1,2,3,4,5,6,7)
```

Pri vyššie uvedenom volaní funkcie functionN, budú teda jednotlivým parametrom zodpovedať argumenty nasledovne:

- parameter1 = 1
- parameter2 = 2

- rest = [3,4,5,6,7]

Spred

Spred operátor možno považovať za opačný k operátoru rest, zatiaľ čo ten zhromažďoval zostávajúce argumenty do poľa, operátor spred „rozbíja“ pole na jednotlivé elementy.

Mohli by sme ho teda použiť napríklad pri zadávaní argumentov našej funkcie functionN v prípade ak by tieto argumenty boli v poli, a to takto:

```
let arguments=[1,2,3,4,5,6,7]
functionN(...arguments)
```

Keďže operátor spred „rozbíja“ pole na jednotlivé elementy často sa využíva aj ku konštrukcii polí na báze už existujúcich polí. Majme teda pole array1 a predstavme si, že chceme vytvoriť nové pole, ktoré bude v podstate len rozšírením poľa array1 o 3 nové prvky, môžeme to teda zapísať nasledovne:

```
let array1 = [1,2,3]

let newArray = [...array1,4,5,6]
```

Keďže operátor spred „rozbíja“ pole na jednotlivé elementy tieto dva príkazy konštruujú rovnaké pole

```
let newArray = [1,2,3,4,5,6]
```

```
let newArray = [...array1,4,5,6]
```

Super článok k rozdielu medzi rest a spred operátorm: https://www.freecodecamp.org/news/javascript-rest-vs-spread-operators/#:~:text=The%20main%20difference%20between%20rest,expands%20iterables%20into%20individual%20elements.

JS tutoriál k rest a spred operátom https://javascript.info/rest-parameters-spread

JSON

JavaScript Object Notation je všeobecný formát na reprezentáciu hodnôt a objektov, ktorý sa používa na čítanie údajov z webového servera a následné zobrazenie týchto údajov na webovej stránke. Keď si vymieňame údaje medzi prehliadačom a serverom, môže to byť iba text, JSON je teda doslova len reťazec.

Syntax JSON reťazca je odvodená z JS objektov používa zložené zátvorky a dvojice kľúč: hodnota (key: value pairs) oddelené čiarkami, no hlavný rozdiel spočíva v tom, že kľuče sú zapísané v úvodzovkách (nie apostrofoch alebo backticks!).

JavaScript poskytuje metódy:

- JSON.stringify pre konverziu JS objektu na JSON
- JSON.parse pre konverziu JSON späť na JS objektu

JSON.stringify

Metóda JSON.stringify() zoberie JS objekt, ktorý jej dodáme ako argument a konvertuje ho na reťazec (JSON). Výsledný reťazec sa nazýva aj JSON-encoded/serialized/stringified/marshalled objekt.

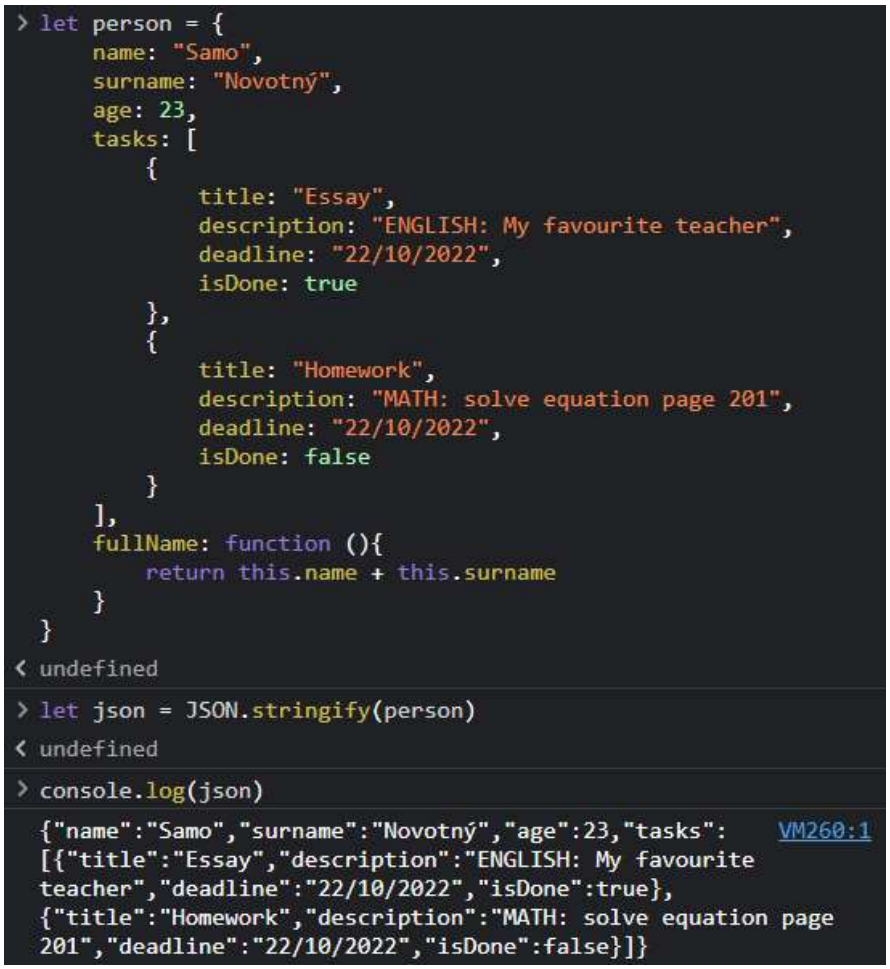
JSON podporuje nasledujúce dátové typy:

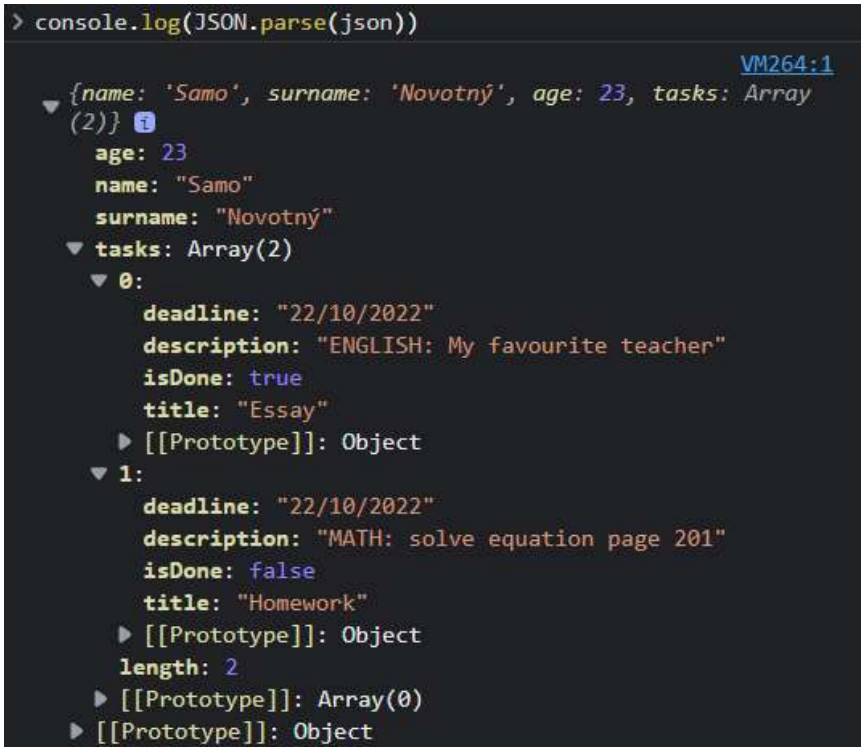
- Objects { ... }
- Arrays [...]
- Primitives:
    - strings,
    - numbers,
    - boolean values true/false,
    - null.

V prípade akJS objekt obsahuje vlastnosti iných typov (function, symboli, undefined) výsledný string nebude obsahovať ich reprezentáciu.

JSON.parse

Metóda JSON.parse() zoberie reťazec (JSON), ktorý jej dodáme ako argument a konvertuje ho na JS objekt. Ide teda o metódu opačnú k metóde JSON.stringify(). Väčšinou ju teda využívame po načítaní dát zo servera.





Super článok k JSON: https://javascript.info/json

Fetch()