Syksy 2020 Harjoitustyö

Testisuunnitelma JavaScript kirjastolle

COMP.SE.200-2020-2021-1 Software Testing

Patric Kangasmäki 428678 - Juho Vasenius 90589

16.10.2020

Esittely

Dokumentti esittelee nettikaupan JavaScript kirjaston testaamista. Kirjasto sisältää erilaisia apufunktioita, joita voi hyödyntää kyseisessä kontekstissa. Sen laajuuden ansiosta sitä voidaan käyttää myös tulevaisuudessa muissa vastaavanlaisissa ohjelmissa. Dokumenttimme kuitenkin pääosin keskittyy testaamaan kirjaston ominaisuuksien toimintaa nimenomaan esitellyssä nettikaupassa.

Dokumentti sisältää testauksen suunnitelman, käytetyt välineet sekä esittelyn testeistä, jotka tulemme toteuttamaan. Dokumentin tarkoituksena on olla alustava pohjapiirros tulevalle testaukselle, jonka toteutamme edellä mainitulle kirjastolle.

Testauksen tavoitteena on varmistaa, että kirjasto toimii oletetulla tavalla ja ylipäätään saada tarkempaa ymmärrystä sen toiminnasta. Testauksen suunnittelussa on apuna kirjaston lähdekoodi, jota on hyödynnetty testien suunnittelussa (White-box testing).

Kirjasto: https://github.com/tgcslearningtech/COMP.SE.200-2020-assignment

Työvälineet

Jest – Valitsimme Jestin testaamiseen, sillä sitä pidetään tällä hetkellä yhtenä parhaimmista testausvälineistä ja ryhmässämme löytyi jo kokemusta sen käyttämisestä. Myös Jestin yksinkertaisuus ja nopeus ovat erinomaisia. Tärkeänä ominaisuutena on myös laadukkaat virheilmoitukset, joita Jest tarjoaa jos ja kun, testeissä menee jokin pieleen.

Dokumentaatio: https://jestjs.io/docs/en/getting-started

GitHub – Versionhallinta ja lähdekoodi – Sopii täydellisesti yhteistyöhön projektin aikana ja lopullisen työn palauttamiseen ominaisuuksiensa ansiosta. Voimme yhdessä työstää testejä ja koodeja ilman huolta, että aiheuttaisimme suuria ristiriitoja tai ongelmia suurempaan kuvaan.

Käyttöohje: https://guides.github.com/activities/hello-world/

Testaukseen vaadittavat paketit ja kirjastot selviävät tarkemmin vasta, kun konkreettinen toteuttaminen alkaa. Toteuttamisen aikana on helpompi huomata, mitä ominaisuuksia tarvitsee eri kirjastoista.

Travis CI – Travis on jatkuvan integraation web-palvelu, jota käytetään ohjelmistoprojektien buildaamiseen ja testaamiseen. Sopii hyvin GitHubin kanssa, sillä palvelu on nimenomaan luotu Gittiä käyttäville projekteille.

Nettisivu: https://travis-ci.org/

Coveralls – Saatamme myös käyttää **Jest**:n sisäänrakennettua testikattavuusraporttia.

Nettisivu: https://www.npmjs.com/package/coveralls

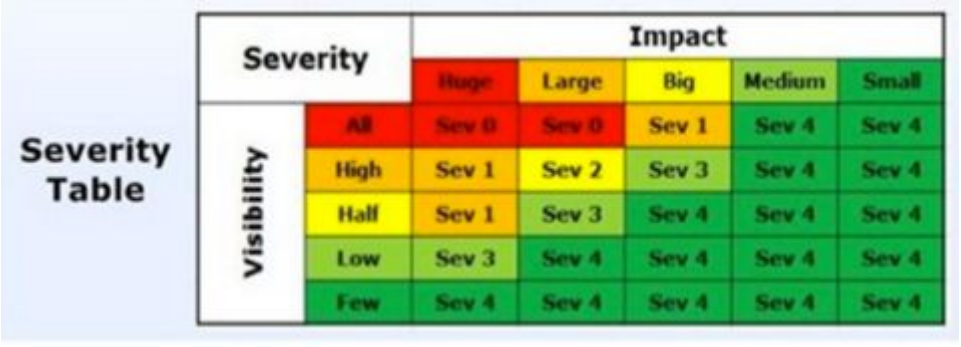
Testitapaukset

Testeissä suoritetaan sekä yksikkö-, että integraatiotestausta. Ensiksi testit luodaan yksittäisille funktioille ja niiden toimivuutta testataan esitetyn sovelluksen käyttötarkoituksissa. Kun funktiot toimivat yksikköinä, siirrytään integraatiotestaamiseen, jossa funktioiden toimintaa tarkastellaan silloin, kun ne ovat toiminnassa muiden osien kanssa.

Testaukset tehdään kirjaston osille, jotka ovat oleellisia esitetyn sovelluksen käyttötarkoituksissa. Jos funktio on täysin epäoleellinen, sen testaaminen sivuutetaan, sillä sen käyttöä eikä toiminnan varmistusta tarvita.

Testien kuvaamiseen käytetään samaa kaavaa. Ensiksi esitellään funktion käyttötarkoitus, jonka jälkeen siirrytään testitapaukseen. Siinä testataan funktioita oikeilla sekä väärillä parametreilla, jonka jälkeen niitä verrataan oletettuun lopputulokseen. Saadut tulokset kirjataan dokumentointiin. Virheellisissä tapauksissa ongelmasta ilmoitetaan kirjaston ylläpidolle.

Jos virheitä ilmaantuu, ne jaotellaan vakavuuden mukaan. Tämän määrittelyyn valitsimme Ivor McCormacin menetelmän, jonka vakavuustaulukon avulla voidaan arvioida ongelman vakavuutta. Menetelmässä virheet luokitellaan niiden vaikutuksen, sekä näkyvyyden mukaan. Luokittelemme testit onnistuneiksi, jos niissä ei ilmene toimintaan vaikuttavia virheitä.



[1] Vakavuustaulukko, jolla virheitä luokitellaan.

Kuvaukset suunnitelluista testitapauksista

camelCase.js

Käyttötarkoitus: Muuttaa parametrina saamansa stringin ‘Camel-case’ muotoon.

Suunnitellut testitapaukset:

- Testataan tyhjällä stringillä, nullilla, undefined, numerolla
- Testataan huomioiduilla erikoismerkeillä
- Testataan jollain muilla erikoismerkeillä
- Testataan useilla välilyönneillä
- Testataan rivivaihdoilla

add.js

Käyttötarkoitus: Laskee yhteen kaksi parametrina saamaansa lukua.

Suunnitellut testitapaukset:

- Testataan nollalla
- Testataan negatiivisilla luvuilla
- Testataan positiivisilla luvuilla
- Testataan tyhjällä stringillä, nullilla, undefined:lla

at.js

Käyttötarkoitus: Pystyt hakemaan objektien sisältä tietueita pathien avulla. Patheja voi olla useita.

Suunnitellut testitapaukset:

- Testataan flattaysta erilaisilla patheilla
- Testataan virheilmoitusta jos käytetään pathia jota ei ole olemassa
- Testataan virheilmoitusta jos pathin syntaksi väärä

capitalize.js

Käyttötarkoitus: Palauttaa parametrina saamansa stringin ‘capitalized’ muodossa.

Suunnitellut testitapaukset:

- Testataan tyhjällä stringillä
- Testataan nullilla
- Testataan undefined
- Testataan numerolla
- Testataan funktiolla
- Testataan objektilla

castArray.js

Käyttötarkoitus: Wrappaa saamansa arvon arrayn sisälle, jos se ei ole jo array, jolloin palauttaa saamansa arvon.

Suunnitellut testitapaukset:

- Testataan, että toimii jos annetaan array.
- Testataan, että toimii jos annetaan jotain muuta.

ceil.js

Käyttötarkoitus: Pyöristää parametrina saamansa arvon ylöspäin toisena parametrina määritettävään tarkkuuteen. (desimaalien määrä)

Suunnitellut testitapaukset:

- Testataan oletustarkkuudella:
    - Testataan negatiiviset
    - Testataan 0
    - Testataan positiiviset
- Testataan positiivisella tarkkuudella
    - Testataan negatiiviset
    - Testataan 0
    - Testataan positiiviset
- Testataan negatiivisella tarkkuudella
    - Testataan negatiiviset
    - Testataan 0
    - Testataan positiiviset

chunk.js

Käyttötarkoitus: Paloittelee taulukon halutun kokoisiin paloihin. Palautetaan taulukkona joka sisältää uudet.

Suunnitellut testitapaukset:

- Testataan suurempaa palastelun kokoa kuin alkuperäisen taulukon koko
- Testataan tavallinen onnistuva case
- Testataan koon 1 palastelu.
- Testataan antaa taulukon sijaan jokin väärä tietomuoto
- Testataan negatiivinen palastelukoko

clamp.js

Käyttötarkoitus: Rajoittaa parametrina saamansa numeron annetulle välille.

Suunnitellut testitapaukset:

- Testataan lukua joka on välin alueella
- Testataan suurempaa kuin välin yläraja
- Testataan välin ylärajalla
- Testataan välin alarajalla
- Testataan pienempää kuin välin alaraja.

compact.js

Käyttötarkoitus: Poistaa falsey elementit taulukosta.

Suunnitellut testitapaukset:

Testataan tavallinen positiivinen testcase (poistaa oikein)

- '0'
- """"
- undefined
- false
- NaN

Testataan negatiivinen case (ei saa poistaa)

- 123
- "foo"
- {}

countBy.js

Käyttötarkoitus: Laskee paljonko collectionissa on objekteja jotka annettuna parametrina parametrina toimitetulle funktiolle, palauttavat true. Käyttää sisäisesti reduce funktiota, hakien jokaiselle objektille tarkastelussa olevan arvon. Koostaa result taulukkoon kuinka paljon näitä arvoja löytyy.

Suunnitellut testitapaukset:

- Testataan collectionilla, jonka koko 1
- Testataan collectionilla, jonka koko 0
- Testataan miten toimii, jos objektilla ei ole pyydettyä jäsentä

defaultTo.js

Käyttötarkoitus: Pitäisi tarkistaa onko ensimmäinen paramteri `NaN`, `null`, tai `undefined` ja palauttaa toinen parametri mikäli näin on. Koodin tarkastelu näyttäisi tsekkavan vain 'null' -tilteen.

Suunnitellut testitapaukset:

Testatan positiiviset caset:

- `NaN`
- `null`
- `undefined`

Testataan negatiiviset caset:

- 123
- "foo"

defaultToAny.js

Käyttötarkoitus: Pitäisi toimia vastaavasti kuin defaultTo, mutta hyväksyy useita arvoja.

Suunnitellut testitapaukset:

- Testataan antaa vain yksi arvo
- Testataan antaa useita arvoja (sisällytetään tähän testiin kaikki defaultTo:n caset)

difference.js

Käyttötarkoitus: Palauttaa taulukon, jossa kaikki arvot jotka olivat uniikkeja ensimmäiselle parametrina toimitetulle taulukolle. Hyväksyy valinnaisen määrän vertailutaulukoita. Päättää samankaltaisuuden 'SameValueZero' spesifikaation mukaisesti.

Suunnitellut testitapaukset:

- Testataan tavanomainen case kokonaisluvuilla
- Testataan tavanomainen case desimaaliluvuilla
- Testataan tavanomainen case objekteilla
- Testataan toimiiko jos mukana "null", "undefined", "NaN"
- Testataan toimiiko jos vertailutaulukko tyhjä

divide.js

Käyttötarkoitus: Toteuttaa jako operaation kahdelle numerolle, palauttaa tarvittaessa desimaaliluvun.

Suunnitellut testitapaukset:

- Testataan jakaa nollalla
- Testataan toinen parametreista numero, toinen string onnistuu
- Testataan molemmat string failaa
- Testataan vain yksi parametri ei kaada

drop.js

Käyttötarkoitus: Palauttaa parametrina saamansa taulukon kopion niin, että sen alusta on poistettu parametrina saatu määrä elementteja.

Suunnitellut testitapaukset:

- Testataan negatiivinen määrä poistettavia
- Testataan tavallinen positiivinen case
- Testataan poistaa enemmän kuin taulukon koko
- Testataan poistaa määrällä 0
- Testataan tyhjällä taulukolla
- Testataan tyhjällä taulukolla ja määrällä 0

endsWith.js

Käyttötarkoitus: Tarkastaa päättyykö parametrina annettu string parametrina annettuun stringiin.

Suunnitellut testitapaukset:

- Testataan tyhjillä stringeillä
- Testataan etsittävä loppuosa pitempi kuin mistä etsitään
- Testataan numeraalisilla parametreilla
- Testataan 'null', 'undefined', 'NaN' parametreilla

eq.js

Käyttötarkoitus: Toteuttaa 'SameValueZero' spesifikaation (http://ecma-international.org/ecma-262/7.0/#sec-samevaluezero) mukaisen tarkastelun kahdelle parametrina toimitetulle arvolle.

Testataan lähdekoodissa dokumentoidut esimerkkicaset:

const object = { 'a': 1 }
const other = { 'a': 1 }

- eq(object, object)// => true
- eq(object, other) // => false
- eq('a', 'a') // => true
- eq('a', Object('a')) // => false
- eq(NaN, NaN) // => true

every.js

Käyttötarkoitus: Saa parametreikseen taulukon ja funktion (predikaatti). Palauttaa; palauttaako jokainen taulukon elementti toimitettuna funktiolle (predikaatti) parametrina true. Funktio (predikaatti) voi ottaa kolme parametria: elementti, sen indeksi, alkuperäinen taulukko.

Suunnitellut testitapaukset:

- Testataan tavanomainen positiivinen case
- Testataan tavanomainen negatiivinen case
- Testataan saako funktio indeksin ja alkuperäisen taulukon parametreinaan
- Testataan tyhjällä taulukolla
- Testataan muuttaako predikaatti alkuperäisen taulukon objekteja

filter.js

Käyttötarkoitus: Saa parametreikseen taulukon ja funktion (predikaatti). Palauttaa alkuperäisen taulukon kopion, josta poistettuna ne elementit jotka palauttivat predikaatille false. Funktio (predikaatti) voi ottaa kolme parametria: elementti, sen indeksi, alkuperäinen taulukko.

Suunnitellut testitapaukset:

- Testataan tavanomainen positiivinen case
- Testataan tavanomainen negatiivinen case
- Testataan saako funktio indeksin ja alkuperäisen taulukon parametreinaan
- Testataan tyhjällä taulukolla
- Testataan muuttaako predikaatti alkuperäisen taulukon objekteja

get.js

Käyttötarkoitus: Palauttaa arvon parametrina saamansa pathin mukaan parametrina saamastaan objektista. Palauttaa tarvittaessa default valuen, joka toimitetaan kolmantena parametrina.

Suunnitellut testitapaukset:

- Testataan väärällä pathin syntaksilla
- Testataan tyhjällä objektilla
- Tavanomainen positiivinen case
- Tavanomainen negatiivinen case

isArguments.js

Käyttötarkoitus: Tarkistaa onko arvo 'arguments' objekti. Epäselvää mihin käytetään, testataan dokumentoinnin esimerkeillä.

Testataan lähdekoodissa dokumentoidut esimerkkicaset:

- isArguments(function() { return arguments }()) // => true
- isArguments([1, 2, 3]) // => false

isArrayLike.js

Käyttötarkoitus: Tarkistaa onko arvo 'array-like'

Suunnitellut testitapaukset:

- Testataan stringillä (pitäisi toimia)
- Testataan taulukolla
- Testataan objektilla
- Testataan kokonaisluvulla (pitäisi palauttaa false)
- Testataan totuusarvolla (pitäisi palauttaa false)

isArrayLikeObject.js

Käyttötarkoitus: Tarkistaa onko arvo array-like ja tarkistaa myös onko tämä arvo objekti. Ei eroa isArrayLike:stä muutoin kun objekti-tarkistukseltaan.

Suunnitellut testitapaukset:

- Testataan stringillä
- Testataan taulukolla
- Testataan objektilla
- Testataan kokonaisluvulla
- Testataan totuusarvolla

isBoolean.js

Käyttötarkoitus: Tarkistaa onko annettu arvo boolean-tyyppinen eli totuusarvo.

Suunnitellut testitapaukset:

- Testataan arvolla True
- Testaaan arvolla False
- Testataan Tyypillä Boolean
- Testaan merkkijonolla
- Testaan objectilla, jolla on boolean arvoja

isBuffer.js

Käyttötarkoitus: Testataan onko annettu arvo Buffer-tyyppinen.

Suunnitellut testitapaukset:

- Testataan antaa merkkijono
- Testataan antaa merkkijonotaulukko
- Testataan antaa integertaulukko
- Testataan antaa objekti
- Testataan antaa buffer

isDate.js

Käyttötarkoitus: Testataan onko annettu arvo Date-tyyppinen

Suunnitellut testitapaukset:

- Annetaan numero-arvo
- Annetaan päivämäärä merkkijonotyyppisenä
- Annetaan päimäärä Date-tyyppisenä

Sovelluksessa käytäntöön: Varmistetaan, että verkkokaupassa näytettävät päivämäärät ovat oikeassa muodossa, jotta ne toimivat varmasti halutulla tavalla.

isEmpty.js

Käyttötarkoitus: Testataan, onko annettu taulukko, objekti tai collection tyhjä.

Suunnitellut testitapaukset:

- Kokeillaan yksittäisiä arvoa
- Kokeillaan taulukkoa
- Kokeillaan objektia
- Kokeillaan merkkijonoa

isLength.js

Käyttötarkoitus: Testataan, voiko annettua arvoa käyttää pituusyksikkönä

Suunnitellut testitapaukset:

- Annetaan merkkijono
- Annetaan int-tyyppinen luku
- Annetaan double-tyyppinen luku
- Annetaan desimaaliluku

isObject.js

Käyttötarkoitus: Testataan, onko annettu arvo object-tyyppinen

Suunnitellut testitapaukset:

- Annetaan objekti
- Annetaan taulukko
- Annetaan funktio
- Annetaan tyhjä arvo
- Annetaan määrittelemätön arvo
- Annetaan merkki
- Annetaan merkkijono

isObjectLike.js

Käyttötarkoitus: Testaan, onko arvo tyypiltänsä object, eikä ole tyhjä arvo.

Suunnitellut testitapaukset:

- Testataan objekti
- Testaan taulukko
- Testataan funktio
- Testataan tyhjä arvo
- Testaan määrittelemätön
- Testataan merkki
- Testataan merkkijono

isSymbol.js

Käyttötarkoitus: Testataan, onko arvo tyypiltänsä symboli tai object symbol

Suunnitellut testitapaukset:

- Testataan Symbolia
- Testataan Objektia
- Testataan yksittäistä merkkiä
- Testataan numeroa
- Testataan merkkijonoa

isTypedArray.js

Käyttötarkoitus: Testataan, onko arvo tyypitetty taulukko

Suunnitellut testitapaukset:

- Testataan merkkiä
- Testataan merkkijonoa
- Testataan taulukkoa
- Testataan tyhjää taulukkoa
- Testataan tyypitettyä taulukkoa

keys.js

Käyttötarkoitus: Luo objektin ominaisuuksien nimistä avain-taulukon

Suunnitellut testitapaukset:

- Annetaan objekti
- Annetaan objekti ilman arvoja
- Annetaan merkkijono
- Annetaan null
- Annetaan undefined

map.js

Käyttötarkoitus: Luodaan uusi lista, joka ottaa annetun funktion alkiot ja käyttää ne halutun funktion läpi ennen uudelle listalle lisäämistä

Suunnitellut testitapaukset:

- Numero-taulukko ja funktio joka kertoo arvot kahdella
- Tyhjä taulukko, toimiva funktio
- Oikea taulukko, rikkinäinen funktio
- Tyhjä taulukko, rikkinäinen funktio
- Oikea taulukko, ei funktiota
- Vääräntyyppiset parametrit

Sovelluksessa käytäntöön: Voidaan käyttää monessa sovelluksen osassa, jossa täytyy käsitellä listoja. Lukemattomia mahdollisuuksia verkkokaupan sivustolla.

memoize.js

Käyttötarkoitus: Luo funktion, joka tallettaa kyseiselle funktiolle annetut arvot prosessin nopeuttamista varten.

Testaan lähdekoodin esimerkit, sekä kokeillaan antaa virheellisiä arvoja ja parametrejä

const object = { 'a': 1, 'b': 2 }

const values = memoize(values)

Suunnitellut testitapaukset:

- values(object)
- values(null)
- values(‘merkkijono’)
- values({})
- values(0)

reduce.js

Käyttötarkoitus: Käytetään annetut kokoelman alkiot annetun supistaja funktion kautta, kunnes jäljellä palautettavaksi on enää yksi arvo.

Käytetään testeissa lähdekoodin antamia reducer-funktio esimerkkejä. Kokeillaan myös antaa vääränlaisia parametrejä ja arvoja.

Vastaanottaa seuraavat parametrit: collection, iteratee, accumulator

Suunnitellut testitapaukset:

- reduce([1, 2], (sum, n) => sum + n, 0)
- reduce([], (sum, n) => sum +n,0)
- Annetaan virheellinen collection
- Annetaan virheellinen iteratee
- Annetaan virheellinen accumulator

Sovelluksessa käytäntöön:

slice.js

Käyttötarkoitus: Poistaa listasta halutun määrän alkioita.

Suunnitellut testitapaukset:

- Toimiva lista, poistetaan listasta alkioita pituuden verran
- Toimiva lista, poistetaan listasta alkioita pituutta enemmän
- Toimiva lista, poistetaan listasta alkioita pituuden vähemmän
- Listan sijaan vääränlainen syöte, poistetaan listasta alkioita oikeellinen määrä
- Listan sijaan vääränlainen syöte, poistetaan listasta alkioita vääränlainen määrä

Sovelluksessa käytäntöön: Voi käyttää esimerkiksi hakutulosten rajaamiseen, Esimerkiksi järjestetään lista hinnan mukaan, sitten poistetaan listasta tuotteita alusta se määrä, kuin tietyn hintaisia tuotteita on. Näin saadaan poistettua esimerkiksi alle 100€ maksavat hakutulokset. Toimii myös toisinpäin. Alle/Yli tietyn hinnan ovat yleisiä toimintoja verkkokaupoissa.

toFinite.js

Käyttötarkoitus: Muokataan annetusta arvosta äärellinen numero

Suunnitellut testitapaukset:

- Desimaaliluku
- Kokonaisluku
- Negatiivinen luku
- Infinity
- Virheellinen merkkijono

toInteger.js

Käyttötarkoitus: Muokataan arvosta integer-tyyppinen

Suunnitellut testitapaukset:

- Annetaan objekti
- Annetaan integer
- Annetaan double
- Annetaan merkkijono
- Annetaan yksittäinen merkki
- Annetaan taulukko
- Annetaan tyhjä taulukko
- Annetaan null
- Annetaan undefined
- Annetaan -0
- Annetaan infinity

toNumber.js

Käyttötarkoitus: Muokataan arvosta number-tyyppinen

Suunnitellut testitapaukset:

- Annetaan objekti
- Annetaan integer
- Annetaan double
- Annetaan merkkijono
- Annetaan yksittäinen merkki
- Annetaan taulukko
- Annetaan tyhjä taulukko
- Annetaan null
- Annetaan undefined
- Annetaan -0
- Annetaan infinity

toString.js

Käyttötarkoitus: Funktio muokkaa annetun arvon merkkijono-tyyppiseksi. Tyhjä ja määrittelemätön palauttavat tyhjän merkkijonon.

Suunnitellut testitapaukset:

- Annetaan objekti
- Annetaan integer
- Annetaan double
- Annetaan merkkijono
- Annetaan yksittäinen merkki
- Annetaan taulukko
- Annetaan json
- Annetaan tyhjä taulukko
- Annetaan null
- Annetaan undefined
- Annetaan -0

upperFirst.js

Käyttötarkoitus: Lauseiden ensimmäiset kirjaimet muutetaan isoksi kirjaimeksi.

Testitapaus:

- Osataan muokata tuotteen kuvaus haluttuun muotoon.
- upperFirst(’laadukas keitin’)
- *Oletettu tulos*: (’Laadukas keitin’)
- upperFirst(’Siisti mikro’)
- Oletettu tulos: (’Siisti mikro’)

Sovelluksessa käytäntöön: Sovellus vaatii, että tuotteiden kuvaukset ovat samantyylisiä, eli kaikissa on ensimmäinen kirjain iso. Funktio soveltuu hyvin tähän.

words.js

Käyttötarkoitus: Käytetään kirjainten vertaamista aakkosiin. Funktio valitsee lähetetystä merkkijonosta aakkosia sisältävät sanat ja laittaa ne omaan taulukkoonsa.

Suunnitellut testitapaukset:

- Annetaan merkkijono ilman erikoismerkkejä
- Annetaan merkkijono erikoismerkeillä
- Annetaan tyhjä merkkijono
- Annetaan merkkijono, jossa pelkkiä erikoismerkkejä
- Annetaan taulukko, jossa merkkijonoja
- Annetaan taulukko, jossa merkkijonoja sekä erikoismerkkejä sisältäviä sanoja

Sovelluksessa käytäntöön: Hakukenttään voidaan kirjoittaa erilaisia hakuehtoja ja funktio osaa eritellä ne sanoiksi, joita pystyy vertaamaan tietokannasta löytyviin tuotteisiin.

Viitteet

Materiaalia, jota käytettiin apuna dokumenttia varten

https://jestjs.io/

https://github.com/

https://www.softwaretestinghelp.com/

https://developer.mozilla.org/fi

[1] https://i.stack.imgur.com/gWLJA.jpg - Ivor McCormacin menetelmä