

Juho Kanninen
Tietojenkäsittelyn koulutusohjelma
Tietojärjestelmät
DBA-kurssin osan AHOT-suoritus
2016

JSON & Tietokannat

“Tämä työ on omistettu opintojeni ohjaajille Anne Jumppaselle, sekä Tuomo Helolle, jotka ovat kärsivällisesti odottaneet valmistumistani Tietojenkäsittelytieteen tradenomiksi. Tämä kärsivällisyys on mahdollistanut ammatillisen kasvuni omalaatuiseksi ja omaehtoiseksi asiantuntijaksi, jolla on kokemusta käytännön ongelmien ratkaisusta ja työkokemusta alalta jo ennen opiskeluiden loppuun saattamista. Tämä omistusteksti on vitsi liittyen pitkään venyneisiin opintoihini.” -Juho Kanninen, jonka vuoden -09 alkanutta opetusohjelmaa ei ole enää olemassa.

Johdanto

Tämän kirjallisen työn on tarkoitus opastaa lukijansa JSON:in käyttöön. Työ vastaa seuraaviin kysymyksiin:

- Mikä on JSON?
- Mihin sitä käytetään?
- Kuinka sitä käytetään?
- Mitä ongelmia siihen liittyy?
- Mitkä ovat JSON:in käytöstä saatavat hyödyt?
- Miten JSON ja tietokannat liittyvät toisiinsa?

Jotta koodi olisi helpommin erotettavissa normaalista leipätekstistä, kaikki koodi on työssä kursivoitu.

Tämän työ olettaa, että lukija osaa perusteet verkkosivujen luonnista ja olio-ohjelmoinnista. Käytetyt ohjelmointikielet: HTML, CSS, JavaScript, PHP ja SQL. Työ sisältää myös englanninkielisiä otteita lähdeoteoksista.

Hyviä termejä osata:

- **Parsing:** Suomalaisittain "Parsea" = Jakaa tiedosto tai syöte helposti manipuloitavaan ja -säilöttävään muotoon. Lauseessa: "XML on raskaampi parsea kuin JSON."
- **Array:** Olio-ohjelmointikielissä käytettävä Lista-olio, jonka sisälle voidaan tallentaa tietoa.
- **API:** Ohjelmointirajapinta (Application Programming Interface), Verkkopalveluun pystytetty osio josta voi pyytää palvelun tietoja oman ohjelman käyttöön.

Tätä työtä kirjoittaessa olen kiinnittänyt erityistä huomiota materiaalin hyödyllisyyteen ja käytännöllisyyteen todellisten ongelmien ratkaisussa, sekä dokumentin rooliin ohjelmointi-opin kartuttamisessa. Linkit ovat sisälletty tekstiin sellaisenaan, jotta opiskelija pääsee suoraan asiasta kertovasta tekstistä lähteeseen, eikä tarvitse käyttää kömpelösti lähdeluetteloa.

1. Mikä on JSON?

Aloitetaan tämä työ nyt tällä pakollisella teorialla. Tämä on se kaikkein epämiellyttävien osa aiheesta opiskelua, mutta on valitettavan välttämätöntä, mikäli et itse halua käyttää tuntejasi googlettamalla JSON:in ihmeellisestä maailmasta. Tein sen tätä työtä tehdessäni sen sinun puolestasi, niin että ei tarvitsisi meidän molempien käyttää arvokasta aikaamme turhiin jaaritteluihin ja kryptisiin asian vierestä selittäviin nettiteoksiin. Tässä työssä tiivistän mielestäni JSON:in käyttämiseen tarvittavat tärkeimmät lähteet ja tiedot. Olen pyrkinyt tekemään tästä dokumentista mahdollisimman käytännönläheisen ja hyödyllisen sinun opinnoillesi.

Noniin aloitetaan!

JSON on lyhenne sanoista **J**ava**S**cript **O**bject **N**otation. Kuten nimestä tulee ilmi JSON:ia käytetään viittaamaan JavaScript-olioihin. JSON on suunniteltu tiedonvälitystä varten ja on ohjelmointikielestä riippumaton avoimen standardin tiedostomuoto. Kieltä käytetään suurelta osin korvaamaan AJAX:in käyttämä XML. (<http://www.json.org>, <http://www.w3schools.com/json/default.asp>)

Miksi minun tulisi osata käyttää JSON:ia?

Lähes kaikki modernit verkkopalvelut ja niiden tarjoamat ohjelmointirajapinnat (API:t) tuottavat tuloksensa JSON-formaatissa. Tästä hyvänä modernina esimerkkinä on Kaupparekisterin kuulutustiedot, jotka on vastikään avattu kehittäjille (<http://avoindata.prh.fi/tr.html#>). Tämän lisäksi on olemassa paljon muita palveluita, joista hakea tietoa ohjelmasi tai verkkosivusi käyttöön. Esimerkiksi säätiedot JSON-muodossa (http://openweathermap.org/current#current_JSON).

Mikäli haluat käyttää tietoja näistä avoimista lähteistä sivustossa tai ohjelmassa, täytyy sinun tuntea JSON:ia ja osata käyttää sitä.

JSON syntaksi eli miten JSON rakentuu ja miten sitä käytetään.

JSON objekti aloitetaan aaltosulkeilla " { " ja päätetään sulkemalla aaltosulkeet " } ". Sulkeiden sisällä määritellään ensin objektin nimi, jonka jälkeen erotetaan nimi sen sisältämistä arvoista kaksoispisteellä. Esimerkki yksinkertaisesta JSON objektista:

```
{ "esimerkkiObjekti" : "EsimerkkiArvo" }
```

JSON objektit voivat myös sisältää useamman nimi ja arvoparin. Nämä erotetaan syntaksin mukaan pilkulla.

```
{ "esimerkkiObjekti1" : "EsimerkkiArvo1!", "esimerkkiObjekti2" : "EsimerkkiArvo2" }
```

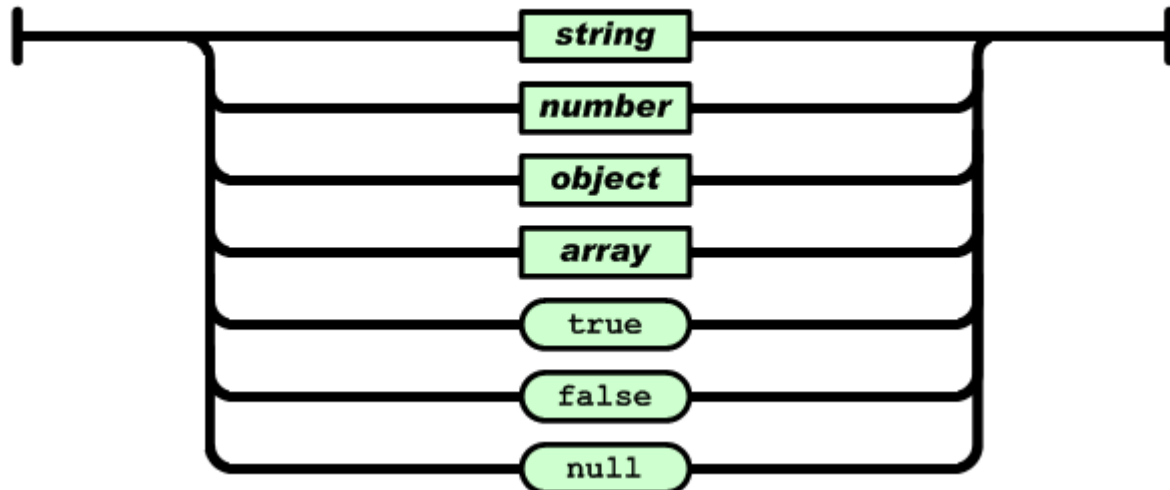
JSON objektit voivat sisältää myös Array elementin jonka sisällä JSON-objektin arvoja säilytetään. Tämä on suuri etu verrattuna vanhaan XML-standardiin. Array määritellään hakasulkeiden sisään "[]" Elementin sisällä noudatetaan JSON:in edellä mainittua syntaksia.

```
{ "tyontekijat": [
    { "etuNimi": "Jukka", "sukuNimi": "Palmu" },
    { "etuNimi": "Aku", "sukuNimi": "Ankka" },
    { "etuNimi": "Mikki", "sukuNimi": "Hiiri" }
  ] }
```

Edellä olevassa esimerkissä Array elementtiin on tallennettu kolme objektia jotka sisältävät työntekijöiden etu ja sukunimen.

JSON objektissa voi käyttää seuraavia arvoja seuraavissa muodoissa:

value



Kuva 1. JSON objektin hyväksymät arvot

2. JSON vs. XML

Nyt tiedämme, miten JSON rakentuu ja tiedämme että se suureksi osaksi on korvannut vanhan XML standardin. Mutta miksi? Onko sellaisia tapauksia joissa JSON:in sijasta tulee käyttää XML:ää? Mitkä nämä tapaukset ovat? Mitkä ovat hyödyt ja haitat kummankin standardin käytöstä?

Haetaanpa näihin kysymyksiin vastauksia. JSON:in puolesta puhuu ei niin yllättävästi verkkosivu nimeltä: (<http://www.json.org/xml.htm>)

Vastaus kysymykseen “Käytänkö JSON:ia vai XML:ää?” löytyykin yllättävän likeltä. Edellä linkitetystä sivulta löysin lauseen “JSON is a better data exchange format. XML is a better document exchange format. Use the right tool for the right job.”.

JSON:in tehokkuuden puolesta puhuvat useat lähteet. Sean Lindo kertoo artikkelissaan, että nimenomaan parempi suorituskyky ja tehokkuus ovat pääsyyt siihen miksi JSON valitaan XML:än yli. (<http://www.programmableweb.com/news/xml-vs.-json-primer/how-to/2013/11/07>) Lindo:n näkökulmasta XML:än luettavuus olisi ihmissilmään JSON:ia parempi, vaikka muut lähteet kuten

JSON.org ja W3Schools puhuvat molemmista vaihtoehtoista itsensä selittävinä ja JSON:ista lyhyempänä ja helpommin kirjoitettuna, sekä käsin että koneellisesti generoituna. Tämän väitteen tukena on W3Schools -sivuston JSON -ohjeesta löytyvä tietokannasta JSON -formaattissa dataa ulos puskeva PHP-koodi. (http://www.w3schools.com/json/json_example.asp)

```
<?php
header("Access-Control-Allow-Origin: *");
header("Content-Type: application/json; charset=UTF-8");

$conn = new mysqli("myServer", "myUser", "myPassword", "Northwind");

$result = $conn->query("SELECT CompanyName, City, Country FROM Customers");

$outp = "[";
while($rs = $result->fetch_array(MYSQLI_ASSOC)) {
    if ($outp != "[") {$outp .= ",";}
    $outp .= '{"Name":"' . $rs["CompanyName"] . '",';
    $outp .= '"City":"' . $rs["City"] . '",';
    $outp .= '"Country":"' . $rs["Country"] . '"}';
}
$outp .= "]";

$conn->close();

echo($outp);
?>
```

Kuva 2. PHP-koodi datan ulos puskemiseen JSON- formaatissa

Hetken tutkailtuasi koodin logiikkaa osaat varmasti itsekkin puskea SQL-serveriltä dataa JSON-formaatissa ulos. Vihdoin jotain käytännön hyötyä tästäkin dokumentista!

Artikkelin viimeisessä kappaleessa Lindo tiivistää syyt formaatin valitsemiseen seuraavalla tavalla:

“There are good reasons for using JSON, and there are still good reasons for using XML. The platform you choose really depends on what you are working to accomplish, the audience and the data that will be shared. XML’s strength is extensibility and the avoidance of namespace clashes. It holds any data type and can be used to transport full documents with formatting information included.

XML is best used when transporting something like a patient chart or text document with markup included. JSON is purposefully limited and therefore much lighter than XML. I suspect that, most of the time, data can be modeled with hashes and lists comprising simple data types, making JSON the preferred route." (<http://www.programmableweb.com/news/xml-vs.-json-primer/how-to/2013/11/07>)

W3Schools ilmoittaa suurimmaksi eroksi JSON:in ja XML:än välillä sen, että XML:ää käytettäessä pitää käyttää omaa XML "parseria". Kun JSON:in parsemiseen tarvitsee puolestaan vain JavaScript-funktion. (http://www.w3schools.com/json/json_intro.asp) Tässä vaiheessa on hyvä kerrata johdannosta mitä parseminen tarkoittaa:

Parsing: Suomalaisittain "Parsea" = Jakaa tiedosto tai syöte helposti manipuloitavaan ja -säilöttävään muotoon. Lauseessa: "XML on raskaampi parsea kuin JSON."

JSON:ia siis kehiin kun kyseessä on puhtaasti datan siirtämisestä ja vastaavasti XML:ää käyttöön kun siirretään tai luodaan dokumentteja joissa formatointi pitää olla kohdillaan. XML:stä löytyy paljon dokumentaatiota ja hyviä tutoriaaleja. Onnekseni tämä on kuitenkin JSON:iin keskittyvä dokumentti, eikä minun tarvitse asiassa mennä sen pidemmälle. Oma kokemukseni dokumenttien koneellisesta generoinnista ei ole nimittäin kovin ruusuinen. Mutta ei siitä sen enempää. Nyt päästään vihdoin asiaan!

3. Verkkosivun yhdistäminen tietokantaan JSON:lla

Tässä kappaleessa tarkastelemme JSON:in käyttöä käytännön esimerkin kautta. Näin saat itse kosketuksen siihen, millaista on työskennellä JSON:in kanssa. Koska olen anteliaalla päällä, saat lisäksi pohjan minkä tahansa ohjelmointiin liittyvän ongelman ratkaisuun.

CASE:

Olet töissä uusien yritysten kirjanpitoa ja raportointia tuottavassa yrityksessä.

Näet että liiketoimintanne perustuu uusien yritysten auttamiseen yrityksensä alkutaipaleella. On siis tärkeää, että olemme perillä niistä yrityksistä, jotka ovat rekisteröityneet kuluvan kuukauden aikana. Meillä on markkinoilla muutama kilpailija, joita ennen meidän tulee löytää potentiaaliset asiakkaamme. Nopeus on valttia uusien asiakkaiden kontaktoinnissa ja hankinnassa. Tarve tavalle löytää nämä uudet rekisteröidyt yritykset on noussut esille johtoryhmän kokouksessa.

Sinut pistetään vastuuseen uuden, tehokkaan kontaktointi tavan kehittämisestä.

Luomme tässä kappaleessa omalle liiketoiminnallemme sisäisen palvelun, jonka toiminta yhdistetään kaupparekisterin tietokantaan. Palvelun avulla pitäisi helposti saada tietoon kuluvan kuukauden aikana rekisteröityneet yritykset. Toimeksiannolla on kova kiire, eikä ulkoasulla ole niin väliä, tärkeää on se, että sovellus toimii moitteettomasti. Ulkoasua voi muokata sitten jälkikäteen paremmaksi.

Haasteena on se, että et ole aikaisemmin liittänyt mitään sovellusta tietokantaan. Siispä teet sen mitä aloitteleva ohjelmistoasiantuntija tekee parhaiten: Ottaa koneen näppäriin kätösiinsä ja alkaa virkein mielin googlettamaan hullun lailla ratkaisua ongelmaansa.

Näin toimit:

Muistat lukeneesi hauskan ja mieleenpainuvan JSON-oppaan, joka sisälsi toimivan listan minkä tahansa ohjelmointiin liittyvän ongelman ratkaisemiseen:

1. **Määrittele ongelma (Mikä on haluttu lopputulos)**
2. **Tutki ongelman olemassa olevia ratkaisuja (Googleta niin maan p*****sti)**
3. **Filtteröi sinulle sopivat ratkaisut (Oikea ohjelmointikieli / Ongelman samankaltaisuus)**
4. **Testaa ja kokeile itse (Niin se vaan on, että tekemällä oppii)**
5. **Jos jäät jumiin, kysy neuvoja! (Kollegat, osaavat ystävät tai nettipalstat)**



Tämä on sykli jota noudatat lukemattomia kertoja ohjelmointiurasi aikana. Koita siis painaa tämä tapahtumasarja mieleesi.

Tätä työtä kirjoittaessani olen käyttänyt yllä olevaa listaa, ja se vain toimii lähes minkä tahansa ongelman kanssa.

Jokaisen vaiheen välissä on tärkeää tutkailla tarvitseeko tilata pizza / keittää kahvia. Myös liikunta ja raikas ilma auttavat varsinkin viidennen vaiheen ehdon täytyessä.

Aloitetaanpa sitten käymällä listaa läpi:

1.Määrittele ongelma

Tiedonsaanti on liian hidasta. Tiedot uusista rekisteröidyistä yrityksistä pitäisi saada nopeammin käyttöön yrityksemme myyjille, jotta kontaktointi tapahtuisi nopeammin.

2.Tutki ongelman olemassa olevia ratkaisuja

Ohjelmointi on siitä hyvä ala, että todella harvassa tapauksessa pyörää täytyy mennä keksimään uudelleen. Suurin osa ongelmien ratkaisuksista on jo saatavilla verkosta, kyse on vain ratkaisujen tehokkaasta löytämisestä ja oikeasta käytöstä. Muistat että Kaupparekisteri on juuri avannut dataansa ohjelmointirajapinnan, mutta et osaa vielä käyttää sitä. Huomaat että API puskee ulos dataa JSON-formaatissa. Siispä sinun täytyy opiskella JSON:in käyttöä.

Huomaat pian että kaikkein käyttökelpoisin lähde on W3Schools:in JSON osio joka löytyy osoitteesta: (<http://www.w3schools.com/json/default.asp>)

Sivustolla oleva tieto on selkeästi lajiteltua ja sisältää jopa testattavaa koodia. Itse asiassa sivustolta löytyvä "JSON Example" sisältää valmiin koodin siihen, miten JSON syötteen saa esitettyä taulukkomuodossa! (http://www.w3schools.com/json/json_example.asp)

3. Filtröi sinulle oikeat ratkaisut

Testaat koodia sivuston omalla työkalulla ja huomaat että tämä ratkaisu sopii täydellisesti jo nykyään Excel-taulukoista tietonsa lukeville myyntihenkilöille.

4. Testaa ja kokeile itse

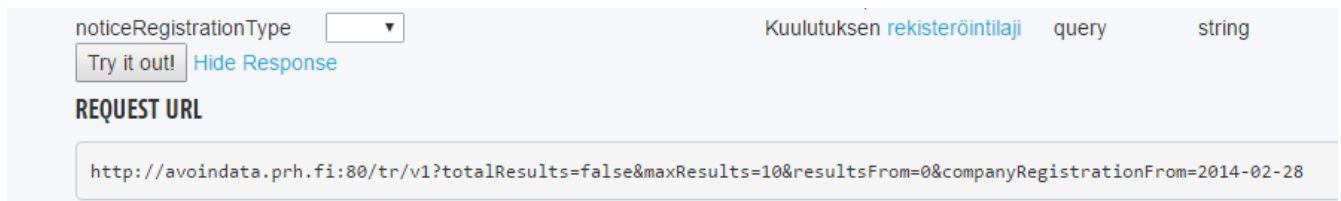
Rupeat pätkäilemään, miten kyseistä koodia pitää muokata, jotta saat sen yhdistettyä kaupparekisterin tietoihin. Huomaat että hiukan alempana on valmiiksi koodia, joka sisältää myös taulukolle hiukan CSS-koodia taulukon helppolukuisuutta auttamaan. Kopioit koodin omaan tekstieditoriisi (Tee tämä nyt!).

Rupeat kommentoimaan koodia, jotta ymmärrät paremmin sen rakenteen ja toiminnan. Saatat nimetä muutaman funktion tai olion uudelleen, jotta koodi on sinulle toiminnaltaan selkeämpi. (Kommentoi oma koodisi siten, miten koodin itse ymmärrät)

```
20     background-color: #ffffff;
21 }
22 </style>
23 </head>
24
25 <body>
26
27 <!-- Varataan alue jolle data tuodaan -->
28 <div id="datakentta"></div>
29
30 <!-- Alla skripti jolla haetaan tiedot palvelusta -->
31 <script>
32
33 //Haetaan URL:in takaa JSON muodossa oleva vastaus
34 var xmlhttp = new XMLHttpRequest();
35 var url = "http://avoindata.prh.fi:80/tr/v1?totalResults=false&maxRes
36
37 //tarkastetaan että saadaan vastaus palvelusta
38 xmlhttp.onreadystatechange=function() {
39     if (xmlhttp.readyState == 4 && xmlhttp.status == 200) {
40         taulukkoKone(xmlhttp.responseText);
41     }
42 }
43 xmlhttp.open("GET", url, true);
44 xmlhttp.send();
45
46 // taulukkoKone tekee tiedoista taulukon
47 function taulukkoKone(response) {
48     var arr = JSON.parse(response).results;
49     var i;
50     var out = "<table>";
51
```

Kuva 3. Kommentoi lainaamaasi koodia parantaaksesi ymmärrystäsi

Huomaat että olio "url" on vastuussa tiedonhausta. Lähdet siis metsästämään haluamaasi tietoja tuottavaa URL:ia. Löydät hetken googlettamisen jälkeen Kaupparekisterin API:n osoitteesta: (<http://avoindata.prh.fi/tr.html>). Hetken leikit kyseisen sivun toiminnallisuuksilla, ja kokeilet eri ehtoja. Sitten huomaat, että siihenhän se on. Linkki jota etsit ilmestyi kun painoit "Try It out!"-



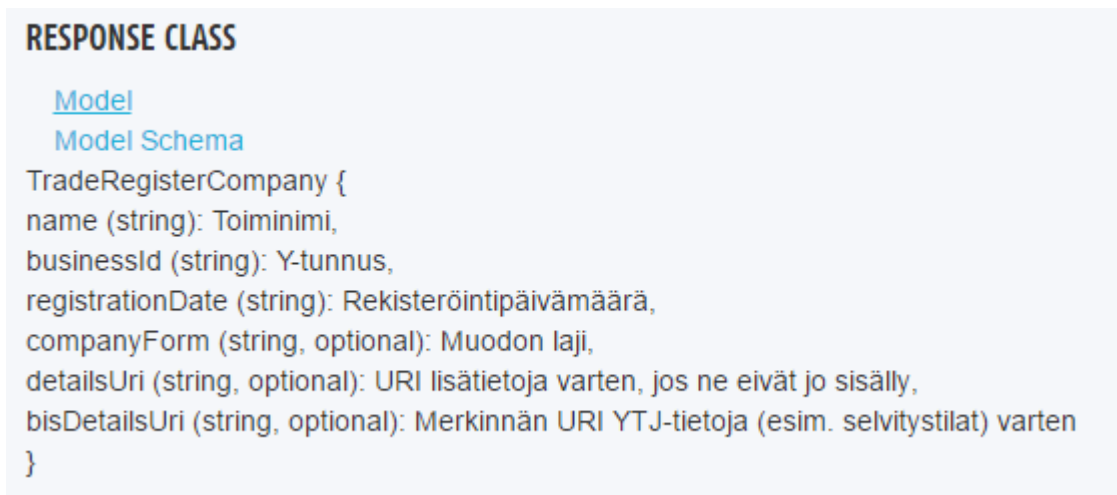
The screenshot shows a REST client interface. At the top, there is a dropdown menu for 'noticeRegistrationType' and a button 'Try it out!'. Below the button, the 'REQUEST URL' is displayed in a text box. The URL is: `http://avoindata.prh.fi:80/tr/v1?totalResults=false&maxResults=10&resultsFrom=0&companyRegistrationFrom=2014-02-28`. Above the URL, there are labels for 'Kuulutuksen rekisteröintilaji', 'query', and 'string'.

Kuva 4. URL löytyi JSON dataan käsiksi pääsemistä varten

painiketta.

Menet tekstieditoriin ja lisäät koodin "url"-olion arvoksi.

Huomaat että W3Sools -sivustolta ottamasi koodin taulukon arvot ei mene yksi yhteen tämän nykyisen tiedon kanssa. Tutkaillet API:n sivulta oikeita arvoja joita etsit sivustolta saamastasi vastauksesta. Löydätkin arvot nopeasti "Model" -osion takaa.



The screenshot shows the 'RESPONSE CLASS' section of the REST client. It displays the 'Model' schema for 'TradeRegisterCompany'. The schema is defined as follows:

```
TradeRegisterCompany {  
  name (string): Toiminimi,  
  businessId (string): Y-tunnus,  
  registrationDate (string): Rekisteröintipäivämäärä,  
  companyForm (string, optional): Muodon laji,  
  detailsUri (string, optional): URI lisätietoja varten, jos ne eivät jo sisälly,  
  bisDetailsUri (string, optional): Merkinnän URI YTJ-tietoja (esim. selvitystilat) varten  
}
```

Kuva 5. Oikeat arvot, joihin viittaaat koodissasi

Vaihdat taulukon arvot koodista oikeisiin arvoihin. Olet sen verran fiksu, että osaat koodista ilmenevää logiikkaa käyttäen lisätä uusia sarakkeita taulukkoon.

Nyt logiikan pitäisi olla kunnossa. Testaat, toimiiko koodi haluamallasi tavalla. Verkkosivu ammottaa tyhjiyytään. Etsit virhettä tunnin tuloksetta. Mikä nyt neuvoksi?

```
for(i = 0; i < arr.length; i++) {  
    out += "<tr><td>" +  
    arr[i].businessId +  
    "</td><td>" +  
    arr[i].name +  
    "</td><td>" +  
    arr[i].registrationDate +  
    "</td><td>" +  
    arr[i].companyForm +  
    "</td><td>" +  
    arr[i].registrationDate +  
    "</td></tr>";  
}
```

Kuva 6. Taulukon luova koodi oikeilla arvoilla

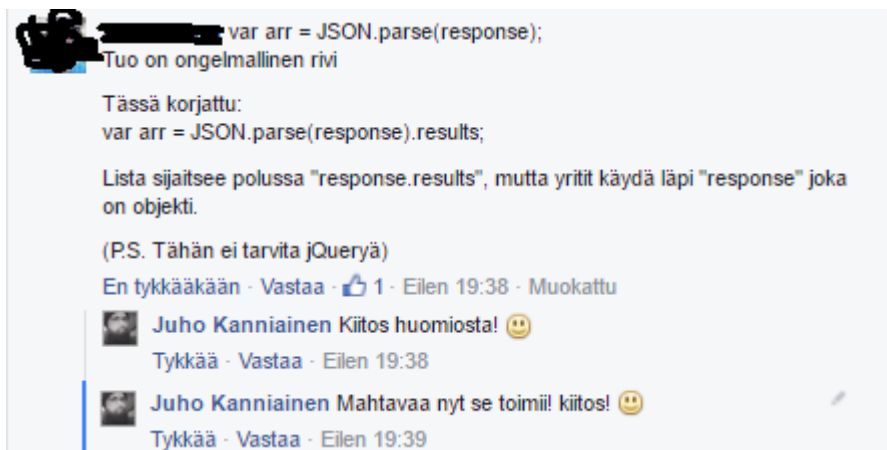
5. Jos jäät jumiin, kysy neuvoja!

Kukaan meistä ei ole niin fiksu, etteikö internetin kollektiivinen viisaus meitä päihittäisi. On siis viisasta käyttää omia kollegoitasi ja muita osaavia ihmisiä apuna. Itse käytin apuna sosiaalisen median ihmeellistä maailmaa. (<https://www.facebook.com/groups/358412240951963/>)



Kuva 7. Apua sosiaalisen median ihmeellisestä maailmasta

Hetken päästä sain kuin sainkin vastauksen! Olin yrittänyt parsea väärää osaa JSON syötteestä!



Kuva 8. Avun pyyntö ei jäänyt ilman vastausta

Pienellä URL:in muokkauksella saa kaikki toukokuun 2015 aikana rekisteröidyt yritykset listattua.

```
"http://avoindata.prh.fi:80/tr/v1?totalResults=false&maxResults=1000&resultsFrom=0&companyRegistrationFrom=2015-05-01&companyRegistrationTo=2015-05-31"
```

Työskennellessäsi hetken aikaa URL:in kanssa, huomaat että sitä muokkaamalla on melko yksinkertaista saada sellaista syötettä kuin haluaa (Testaa itse urlin muokkaamista). JSON syötteen mukana tulee aina linkki seuraaviin tietoihin jotka kyseiset hakuehdot täyttävät. Voit halutessasi tehdä oman URL generaattorin, jolloin vain muuttujan arvoa vaihtamalla pystyt vaikuttamaan suoritettuun hakuun.

```
42 // uusi url generaattori
43 var totalResults= false;
44 var resultsFromCount = 0;
45 var maxResults = 20;
46 var companyRegistrationFrom = "2015-05-01";
47 var compantRegistrationTo = "2015-05-31";
48
49 var url = "http://avoindata.prh.fi:80/tr/v1?
totalResults="+totalResults+"&maxResults="+maxResults+"&resultsFrom="+resultsFromCount+"&companyRegistrationFrom="+companyRegistrationFrom+"&comp
anyRegistrationTo="+compantRegistrationTo;
50
```

Kuva 9. URL-generaattori URL:in koneelliseen manipulointiin

Kokeiltuasi "resultsFromCount" -arvolla leikkimistä, huomaat että muuttamalla arvoa pystyt kontrolloimaan monenneltako tietueelta lista alkaa. Sinulla välähtää: Tarvitset vain nappulat, joiden onclick metodien sisälle pistät funktion, joka korottaa tai laskee "resultsFromCount"- olion arvoa "maxResults"-olion arvolla! Tämän onnistuminen vaatii vain URL generaattorin sijoittamista omaan funktioonsa, jota kutsutaan nappia painaessa, jotta URL generoitaisiin uudelleen. Tällä tavalla voit selata JSON-syötettä, eikä sinun tarvitse tulostaa kaikkia rivejä yhdelle sivulle. Sen jälkeen sinun tarvitsee vain alustaa aika sillä tavoin, että haku palauttaa aina meneillään olevan kuukauden aikana rekisteröidyt yritykset. (Valmis koodi osoitteesta:

<https://github.com/JuhoKanniainen/OpenFinCorpTableMaker/blob/master/FinnCorpOpenDataTableMaker.html>

Sovellus on nyt valmis ja myyjät pääsevät kiinni uusimpiin rekisteröityneihin yrityksiin kiitos sinun!

4. Loppusanat

Noniin! Olemme päässeet tämän mahtavan ja opettavaisen dokumentin loppuun. Kysy siis itseltäsi, osaatko vastata seuraaviin kysymyksiin:

- Mikä on JSON? (kappale 1)
- Mihin sitä käytetään? (kappaleet 1 & 2)
- Kuinka sitä käytetään? (kappaleet 1 & 3)
- Mitä ongelmia siihen liittyy? (kappale 2)
- Mitkä ovat JSON:in käytöstä saatavat hyödyt? (kappale 2)
- Miten JSON ja tietokannat liittyvät toisiinsa? (kappaleet 2 & 3)

Mikäli osaat näihin kysymyksiin vastata, niin tämä dokumentti on tehnyt tehtävänsä. Kaikki ylimääräinen mitä olet tästä dokumentista oppinut, on plussaa. Tästä on hyvä lähteä eteenpäin kehittämään erilaisia JSON:ia hyödyntäviä sovelluksia!

Tarkoituksenani oli luoda yleishyödyllinen ja oikeasti käytännön suorittamisessa auttava työ. Dokumentti, joka ei ole pelkkää teoriaa ja sitoo opetettavan aiheen tosielämän haasteisiin. Mielestäni onnistuin siinä, mutta erikin mieltä saa olla. En tiennyt ennen tämän työn tekemistä JSON:sta mitään, mutta nyt voin ylpeänä todeta, että tiedän yhtä paljon kuin tämän dokumentin orjallisesti läpi lukenut henkilökin!

Summa summarum: Tietokannat sisältävät tietoa. JSON on kevyt ja helposti kirjoitettava/-ymmärrettävä/-generoitava formaatti siirtää sitä eri ohjelmien välillä.

Ai niin! ProgrammableWeb:in sivuilta löydät rapiat 15 000 eri ohjelmointirajapintaa käytettäväksesi. Että siinä sinulle valinnanvaraa! (<http://www.programmableweb.com/category/all/apis>)

Kiitos ja kumarrus:

Juho Kanniainen (<https://sumry.me/juhokanniainen>)

Tietojenkäsittelyn koulutusohjelma

Tietojärjestelmät

-09 Opetussuunnitelma, jota ei enää ole olemassa.

5.Lähteet

Apuja Facebookista (Ohjelmointi-ryhmä) :

<https://www.facebook.com/groups/358412240951963/>

JSON syntaxin virallinen dokumentaatio:

<http://www.ecma-international.org/publications/files/ECMA-ST/ECMA-404.pdf>

JSON syntaxin epävirallinen mutta helpommin aukeava dokumentaatio:

http://www.w3schools.com/json/json_syntax.asp

<http://www.json.org/>

JSON Tutorial (ehdottomasti paras JSON:ia opettava dokumentaatio omasta mielestäni)

<http://www.w3schools.com/json/default.asp>

Oma koodini käytännön osion toteutuksesta (kappale 3)

<https://github.com/JuhoKanniainen/OpenFinCorpTableMaker/blob/master/FinnCorpOpenDataTableMaker.html>

Millä tavoin JSON eroaa XML:ästä?:

<http://www.json.org/xml.html>

Sean Lindon "JSON vs. XML -primer" artikkeli:

<http://www.programmableweb.com/news/xml-vs.-json-primer/how-to/2013/11/07>

Yli 15000 käytettävissä olevaa ohjelmointirajapintaa:

<http://www.programmableweb.com/category/all/apis>