OULUN AMMATTIKORKEAKOULU

Topi Stolp

HTML5-POHJAINEN ÄÄNENNAUHOITUSOHJELMA

HTML5- POHJAINEN ÄÄNENNAUHOITUSOHJELMA

Topi Stolp
Opinnäytetyö
Syksy 2014
Tietotekniikan koulutusohjelma
Oulun ammattikorkeakoulu

TIIVISTELMÄ

Oulun ammattikorkeakoulu
Tietotekniikan koulutusohjelma, ohjelmistokehitys

Tekijä: Topi Stolp
Opinnäytetyön nimi: HTML5-pohjainen äänennauhoitusohjelma
Työn ohjaaja: Veijo Väisänen
Työn valmistumislukukausi ja -vuosi: Syksy 2014
Sivumäärä: 29 + 0 liitettä

Tämän opinnäytetyön aiheena oli luoda HTML5-pohjainen äänennauhoitusohjelma, jota käytetään osana Otometri Oy:n korvatulehdusmittausohjelmistoa. Otometrin tarjoamalla palvelulla vanhemmat voivat seurata lastensa korvien terveyttä kotona. Uudella, pelkästään selaimella toimivalla nauhoitusohjelmalla oli tarkoitus korvata aikaisempi ladattava Java-versio.

Työ aloitettiin perehtymällä HTML5:n tarjoamiin mahdollisuuksiin nauhoittaa ja toistaa ääntä. Mahdollisuuksien toteamisen jälkeen ohjelma suunniteltiin ja toteutettiin. Koska Otometri on kliinisesti tutkittu tuote, uuden mittausohjelmiston kehityksen aikana tuloksia oli verrattava vanhan mittausohjelmiston kanssa, jotta voitiin olla varmoja siitä, että uuden nauhoitusohjelman tulokset ovat luotettavia.

Työ onnistui vaaditulla tasolla ja otettiin käyttöön yrityksessä sen valmistuttua. Koska HTML5 on vielä kehittyvä tekniikka, eikä se ole täysin tuettu, myös vanha Java-versio jätettiin valittavaksi vaihtoehdoksi suorittaa mittaus. Uusi HTML5-versio mahdollistaa myös jatkokehityksen, jolloin mittauksen voi suorittaa tulevaisuudessa myös mobiililaitteilla.

Asiasanat: HTML5, äänentallennus, korvatulehdus, itsehoito

ABSTRACT

Oulu University of Applied Sciences
Information Technology, software development

Author: Topi Stolp
Title of thesis: HTML5-based audio recorder program
Supervisor: Veijo Väisänen
Term and year when the thesis was submitted: Fall 2014
Pages: 29 + 0 appendices

The goal for this thesis was to create a new audio recorder program which runs solely on browser with HTML5. The thesis was assigned by Otometri Ltd which offers a service for parents where they can measure their children ears at home and monitor for ear infections.

I started this project by checking the possibilities of HTML5 regarding audio playback and recording. After confirming the needed features, I designed and implemented the old Java-based program in HTML5. During the implementation I had to check that the new HTML5 version would give similar audio files and results as the Java version.

The project goal was achieved and the new HTML5 audio recording program was taken into use after it was finished. Because HTML5 is still in development and not supported on every single browser, the old Java version was kept alongside the new HTML5 version so users have the ability to use the Java version if they want or can't use the new HTML5 version. HTML5 gives also possibilities for future development for mobile support.

Keywords: HTML5, audio recording, ear infection, self care

SISÄLLYS

TIIVISTELMÄ 3
ABSTRACT 4
SISÄLLYS 5
SANASTO 6
1 JOHDANTO 8
2 HTML5 JA OTOMETRIN MITTAUSOHJELMA 12
2.1 HTML5 12
2.2 Web Audio API 12
2.3 Otometrin mittauksen kulku 14
3 NAUHOITUSOHJELMAN SUUNNITTELU 16
3.1 USB-äänilaitteen tunnistaminen 16
3.2 Äänen toistaminen 16
3.3 Äänen nauhoittaminen 17
4 NAUHOITUSOHJELMAN TOTEUTUS 18
4.1 Äänen toistaminen 18
4.2 Painikkeen painalluksen tunnistaminen 21
4.3 Äänen nauhoittaminen 23
4.4 Ääniasetusten säätäminen 24
5 TULOSTEN VERTAILU 25
6 YHTEENVETO 27
LÄHTEET 28

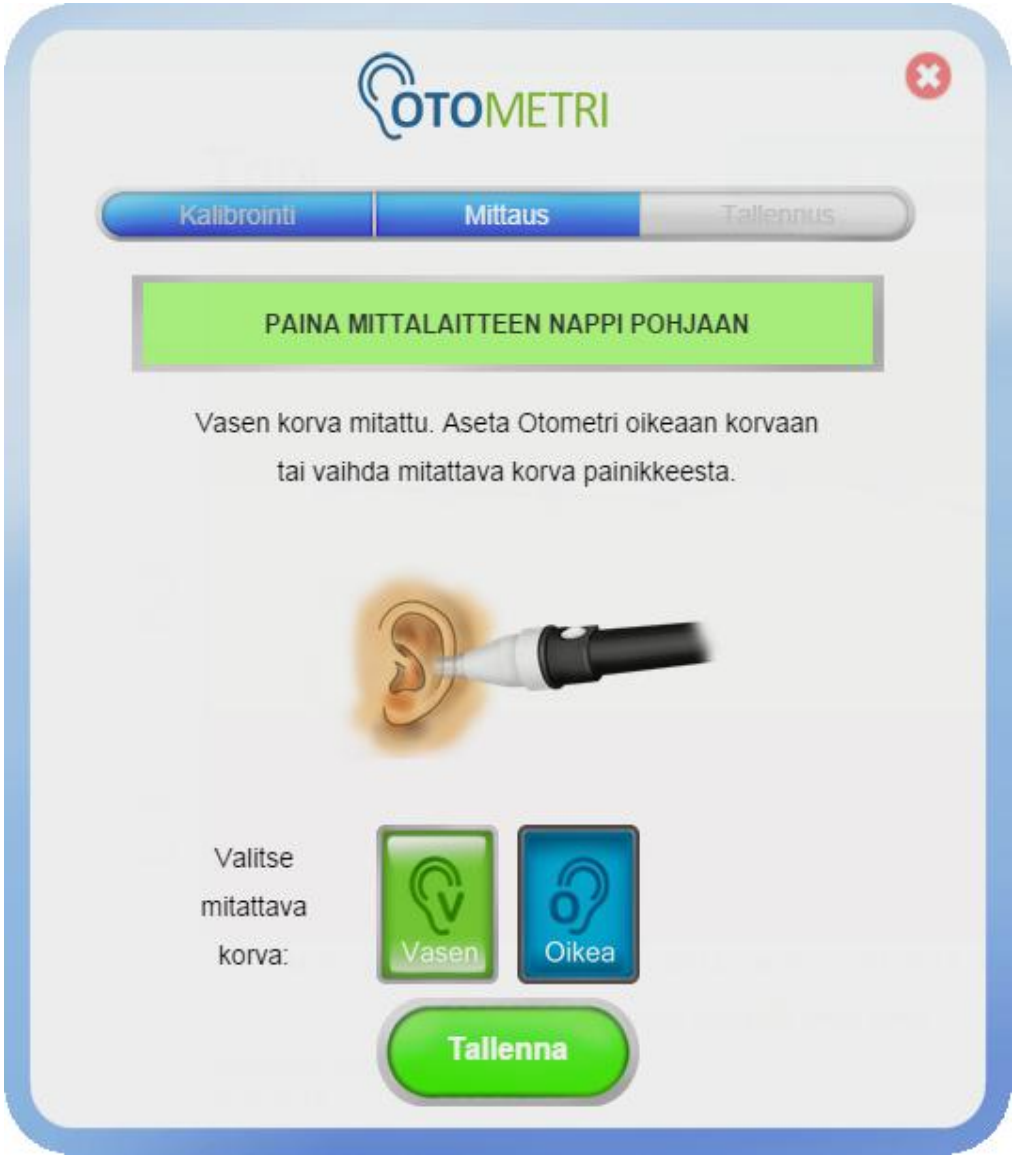
SANASTO

ArrayBuffer	Puskuri, joka pitää sisällään geneeristä binaaridataa. Tätä dataa ei voi suoraan muokata.
Asynkroninen	Ajasta ja paikasta riippumaton tapahtuma. Kun ohjelmoinnissa suoritetaan asynkroninen funktiokutsu, ohjelma ei jää odottamaan funktion vastausta, vaan palaa siihen, kun funktion suoritus on valmis.
Blob	Tiedosto, joka koostuu binaaridatasta. Yleensä kuva-, ääni- tai multimediatiedosto.
HTML5	HyperText Markup Language. Nettisivujen luomiseen käytetty merkintäkielen uusin kehitteillä oleva versio.
JAR	Java Archive. Tiedostomuoto, joka pakkaa Java-ohjelmointikielellä kirjoitetun applikaation suoritettavaksi paketiksi.
Javascript	Verkkosivuilla käytetty komentosarjakieli.
JNLP	Java Network Launching Protocol. Tiedosto, joka määrittelee ladattavan JAR-tiedoston ja muut tarvittavat parametrit XML-muodossa.
Korrelaatio	Arvo, joka kuvaa kahden muuttujan välistä riippuvuutta. Mitä lähempänä muuttujat ovat toisiaan, sitä lähempänä korrelaatio on arvoa 1 tai –1.
MIT-lisenssi	Vapaa ohjelmistolisenssi, joka on kehitetty Massachusetts Institute of Technologyssä. Lisenssi antaa käyttäjälle oikeudet muokata ja myydä tuotetta eteenpäin omassa projektissa.

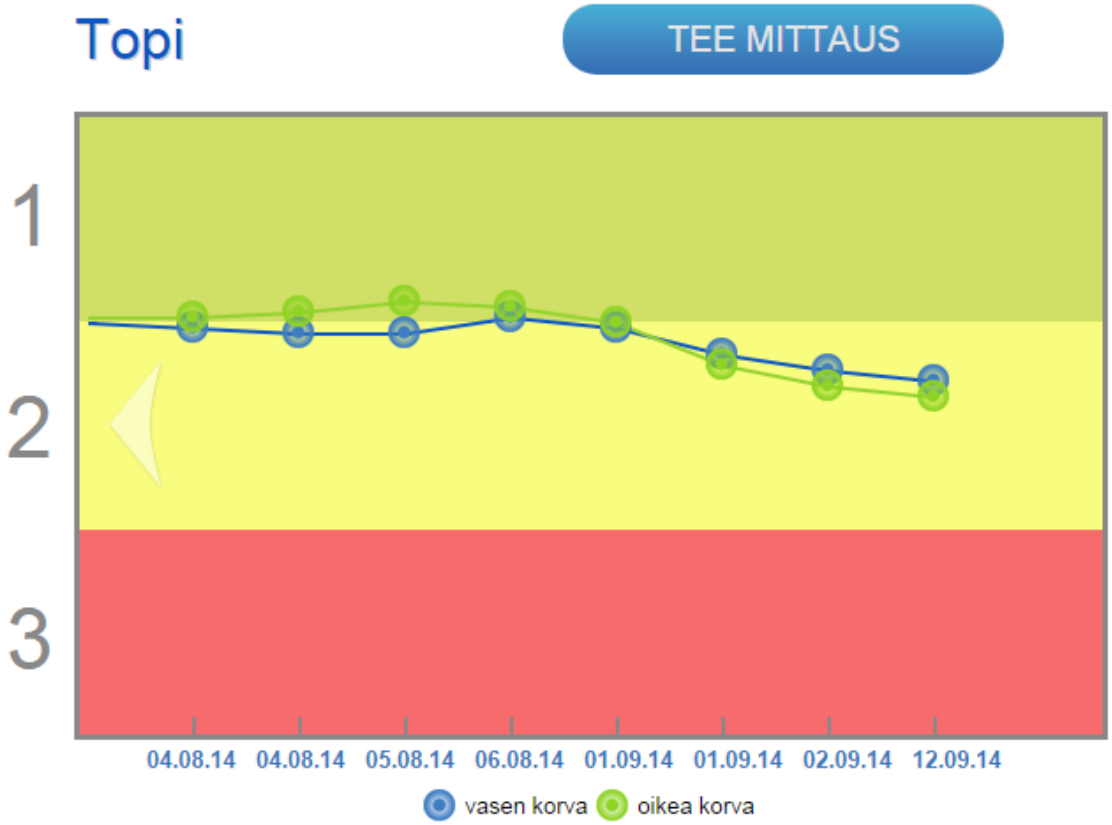
Puskuri	Muistialue, johon kerätty data tallennetaan väliaikaisesti.
Wav	Microsoftin ja IBM:n kehittämä audioformaatti.
XMLHttpRequest	JavaScript-objekti, joka mahdollistaa datan hakemisen URL-osoitteesta ilman sivun uudelleenlatausta.
Äänen spektri	Määritelmä, jossa ääniaalto on jaettu eri taajuisiin osiin.

1 JOHDANTO

Otometri Oy on kempeleläinen yritys, joka tarjoaa palvelun, jonka avulla asiakkaat voivat seurata lastensa korvien terveyttä kotona. Mittauksen käyttöliittymä on toteutettu Otometrin nettisivuille (kuva 1). Mittaustulokset säilyvät tallessa Otometrin palvelimella ja niitä voi selata vapaasti käyttäjätunnusten avulla miltä tahansa laitteelta, jossa on Internet-yhteys (kuva 2).



KUVA 1. Mittausohjelman käyttöliittymä. Vasen korva mitattu onnistuneesti.



KUVA 2. Mittaushistoria Otometrin mittauskeskussivulla.

Mittalaitteena toimii Otometri Oy:n kehittämä tietokoneen USB-porttiin liitettävä laite. Mittalaite toimii erillisenä äänikorttina ja sitä käytetään mittauksessa. Mittalaitteessa on kaiutin, mikrofoni ja painonappi. (Kuva 3.)

KUVA 3. Otometri-tuotepaketti (Hoida korvatulehdusta jo kotona. 2014)

Mittaus perustuu akustisen reflektometrian ja neuroverkon yhdistelmään. Akustisen reflektometrian ensimmäisen mittausperiaatteen esittelivät David W. Teele ja John Teele vuonna 1984. Akustisessa reflektometriassa korvakäytävään lähetetään äänisignaali, joka heijastuu tärykalvosta takaisin ja tämä heijastuma mitataan. Korvatulehduksen aikana välikorvaan syntyy nestettä ja muuta eritettä, mikä vaikuttaa heijastumaan. (Akustinen reflektometria ja neuroverkot Otometrin ytimenä. 2009.)

Nauhoitettu heijastuma lähetetään neuroverkolle analysoitavaksi ja se palauttaa lasketun indeksiarvon mittaukselle. Neuroverkko on mallinnus ihmisen aivoista, joka oppii sille syötetyn datan perusteella. Otometrin käyttämä neuroverkko kehittyykin jatkuvasti mittausten tekemisten yhteydessä, sillä käytettävä opetusaineisto kasvaa aina, kun mittaus tehdään. (Akustinen reflektometria ja neuroverkot Otometrin ytimenä. 2009.)

Tämän opinnäytetyön tarkoituksena on päivittää Otometri Oy:n korvatulehdusmittausohjelma ladattavasta Java-versiosta kokonaan selaimella toimivaksi HTML5-versioksi. Otometrin mittauskeskussivu käyttääkin jo HTML5:n tarjoamaa canvas-elementtiä mittaushistorian seuraamiseen, joten asiakasohjelmiston päivittäminen HTML5-pohjaiseksi on luonnollinen jatkumo.

Mittausohjelmasta haluttiin kokonaan selaimella toimiva versio, sillä Java-pohjaisessa ratkaisussa käyttäjän pitää aina ensin ladata JNLP-tiedosto omalle koneelleen ja koneella tulee olla Java asennettuna, ennen kuin mittauksen voi suorittaa. Java-ohjelmistosta löytyy myös tietoturva-aukkoja useamman kerran vuodessa ja näitä aukkoja korjataan päivityksillä. Mikäli käytössä on vanhempi versio Javasta, on tietokone haavoittuvainen hyökkäyksille. Yleinen ohje onkin estää netistä ladattavien Java-ohjelmien suorittaminen, joten siirtyminen Javasta HTML5-pohjaiseen ohjelmistoon on suotavaa myös asiakkaiden tietoturvan kannalta. (Java-estot parantavat tietoturvaa, toimi näin. 2014.)

Olen ollut itse mukana aikaisemmassa Otometrin ohjelmistokehityksessä kesällä 2012, jolloin Java-ohjelmisto päivitettiin nykyiseen muotoonsa. Näin ollen mit-

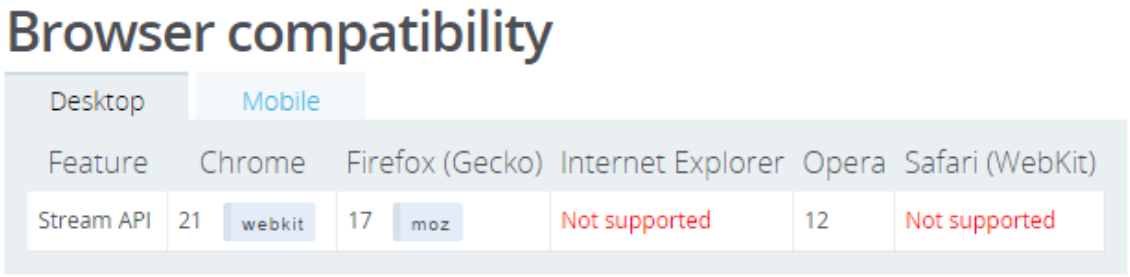
tausohjelman asiakas- ja palvelinohjelmisto olivat ennestään tuttuja ja työn aloittaminen oli helpompaa.

2 HTML5 JA OTOMETRIN MITTAUSOHJELMA

2.1 HTML5

HTML5 on uusin kehitteillä oleva versio HTML-merkkauskielestä, jota käytetään verkkosivujen tekemiseen. HTML5 esittelee uusia elementtejä, kuten video- ja audioelementit, joilla voidaan toistaa videota sekä ääntä nettiselaimella, sekä canvas-elementin, jolla voidaan piirtää 2D- ja 3D-grafiikkaa selainikkunassa.

Mikrofonin käyttöä varten tarvitaan selaimen getUserMedia-funktiota. Funktiolle annetaan vähintään 2 parametria: mitä medialaitetta halutaan käyttää, mikrofonia, webkameraa vai molempia, ja mitä tehdään, jos medialaite on saatu käyttöön. Kun funktiota kutsutaan, käyttäjältä pyydetään lupa halutuiden medialaitteiden käyttämiseen selaimessa. Ohjelman tekohetkellä Internet Explorer ei tukenut tätä funktiota, joten sen toimivuus uudessa HTML5-pohjaisessa mittausohjelmassa oli poissuljettu heti työtä aloitettaessa (kuva 4). Web Audio API:sta on tehty ehdotus Microsoftille ja Internet Explorerin kehittäjät ovat ilmaisseet kiinnostuksensa sitä kohtaan. Oletettavasti tuki tulee versiossa 12. (Web Audio API support. 2014.)



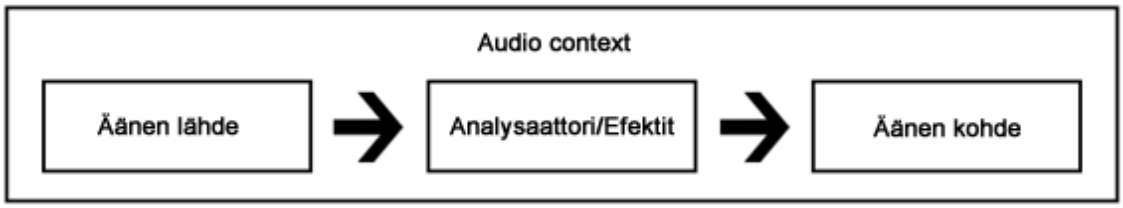
KUVA 4. getUserMedia-funktion tuki eri selaimilla (NavigatorUserMedia.getUserMedia(). 2014)

2.2 Web Audio API

Web Audio API on ohjelmointirajapinta, joka tarjoaa mahdollisuuden äänen käsittelyyn selaimessa. Sen avulla voidaan muun muassa valita toistettava ääni,

lisätä efektejä, luoda audiovisuaalisia elementtejä, analysoida ääntä ja tallentaa ääntä tietokoneeseen liitetyllä mikrofonilla. (Mills 2014.)

Rajapinnan korkein taso on AudioContext-luokka (Adenot – Wilson 2013b). Tästä luokasta luotu olio pitää sisällään kaikki muut äänenkäsittelyssä tarvittavat objektit ja yhdistää ne toisiinsa. Tyypillisessä rakenteessa AudioContext pitää sisällään kolme nodea: äänen lähde, analysaattori/efektit ja äänen kohde. Useimmilla nodeilla on sisääntulo ja ulostulo, joiden kautta ne yhdistetään seuraavaan nodeen. (Kuva 5.)



KUVA 5. Tyypillinen web audion rakenne (Web Audio API. 2014)

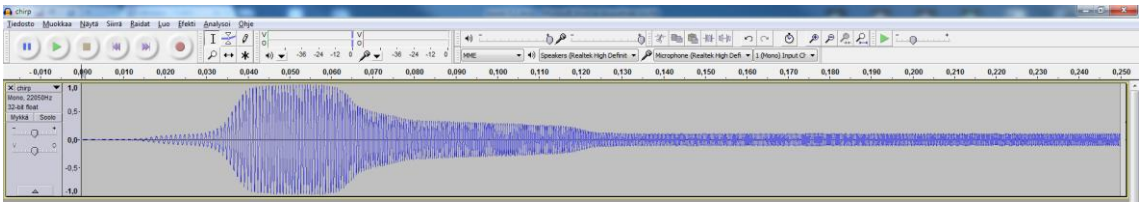
Äänen lähde voi olla esimerkiksi tietokoneeseen liitetty mikrofoni tai erillinen äänitiedosto. Analysaattori on node, joka ottaa vastaan äänen, luo siitä reaaliaikaista taajuus- ja aikapohjaista dataa ja lähettää sen muuttumattomana eteenpäin. Tätä dataa voidaan käyttää äänen tutkimiseen ja visualisointiin. Äänen kohde on esimerkiksi tietokoneen kaiuttimet, jotka toistavat syötetyn äänen.

Mikäli äänen lähteeksi halutaan äänitiedosto, sen luomiseksi tarvitaan rajapinnan tarjoamaa AudioBuffer-luokkaa. Tästä luokasta luotu olio pitää sisällään äänitiedostosta ladatun datan. Äänitiedosto voidaan ladata AudioBuffer-olioon AudioContextin tarjoamalla decodeAudioData-funktiolla. Tämä funktio lataa asynkronisesti äänitiedoston datan ja muuttaa sen AudioBuffer-muotoon, joka voidaan liittää äänen lähteeseen.

Tässä tapauksessa äänen lähde on AudioBufferSourceNode-luokasta (Adenot – Wilson 2013a) luotu olio. Tällä oliolla ei ole sisääntuloa, vaan pelkästään ulostulo. Ulostulo liitetään AudioDestinationNodeen (Adenot – Wilson 2013c), joka on yleensä tietokoneen kaiutin.

2.3 Otometrin mittauksen kulku

Otometrin mittauksessa käytetään ääniaaltoa, joka koostuu useista eri taajuuksista. Näin saadaan aikaan äänenkorkeudeltaan nouseva ääniaalto. Tätä ääniaaltoa kutsutaan ohjelmassa chirpiksi. (Kuva 6.)



KUVA 6. Chirpin ääniaalto Audacity-ohjelmassa.

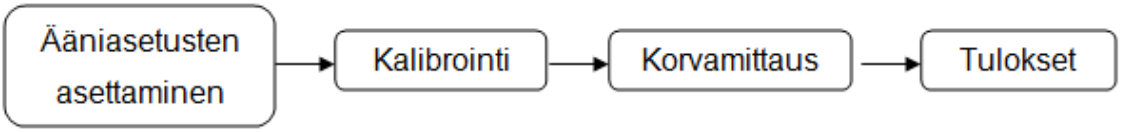
Ääni on aaltoliikettä, joka etenee väliaineessa. Äänen ominaisuuksia ovat mm. amplitudi, eli kuinka voimakkaasti ääni kuuluu, sekä taajuus, eli kuinka korkea kyseinen ääni on. (Ääni. 2014.)

Mittauksen eteneminen tapahtuu lähes joka kerta samalla kaavalla. Ääniasetukset asetetaan kohdilleen, mikä tarkoittaa sitä, että mikrofonin ja kaiuttimen äänenvoimakkuudet ovat oikealla tasolla. HTML5-versiossa ääniasetukset tulee säätää käsin ennen kuin mittausohjelman käynnistää.

Tämän jälkeen käynnistetään mittausohjelma ja suoritetaan automaattinen kalibrointi, jonka aikana mittalaitteessa on korkki paikoillaan. Kun kalibrointi on tehty, mittalaitteen korkki poistetaan.

Tässä vaiheessa mittausohjelma alkaa kuuntelemaan painikkeen painallusta. Kun painiketta on painettu, mittalaite toistaa äänen, nauhoittaa heijastuman ja lähettää sen palvelimelle, jossa neuroverkko laskee indeksiarvon mittaukselle. Mikäli mittaus ei onnistu, antaa käyttöliittymä vastaavan palautteen ja mittausta voi koittaa uudestaan. Kun käyttäjä on saanut onnistuneen mittaustuloksen, tämän jälkeen voidaan mitata sama korva uudelleen, jolloin edellinen tulos poistuu, mitata toinen korva tai mennä suoraan tarkastelemaan mittaustuloksia. (Kuva 7.)

Mikäli käyttäjä aloittaa uuden mittauksen saman selainistunnon aikana, mittausohjelma käyttää edellistä kalibrointia tulosten laskemiseen. Tällöin mittausohjelman käynnistyessä mennään suoraan korvamittauksiin.



KUVA 7. Mittauksen peruskulku.

3 NAUHOITUSOHJELMAN SUUNNITTELU

Koska HTML5 on vielä kehitteillä oleva tekniikka, ennen työn aloittamista oli selvitettävä, missä määrin nykyinen mittausohjelmisto voitaisiin toteuttaa HTML5-tekniikalla. Jotta HTML5:tä voitiin käyttää tässä projektissa, piti sen pystyä tekemään seuraavat asiat:

- tunnistamaan laite ja käyttämään USB-porttiin kytkettyä äänilaitetta
- toistamaan wav-muotoista ääntä
- nauhoittamaan ääntä wav-muotoon.

Lisäksi toivottava, mutta ei pakollinen ominaisuus oli USB-porttiin kytketyn äänilaitteen ääniasetusten automaattinen säätäminen.

3.1 USB-äänilaitteen tunnistaminen

Kun Otometri kytketään tietokoneeseen, käyttöjärjestelmä tunnistaa laitteen automaattisesti ja asentaa tarvittavat ajurit. Jatkossa ajureita ei tarvitse asentaa, jos käyttää samaa USB-porttia. Äänennauhoitusohjelmassa käytettävä koodi ei erikseen valitse käytettävää äänilaitetta, vaan käyttää automaattisesti käyttöjärjestelmän oletusäänilaitetta. Yleensä käyttöjärjestelmä asettaa viimeisimmän kytketyn äänilaitteen oletusäänilaitteeksi, joten laitteen tunnistamisessa ei tule ongelmia.

3.2 Äänen toistaminen

Otometrin mittausohjelmassa käytetään kahta eri äänitiedostoa. Toinen tiedosto on taustakohina, jota käytetään apuna mittalaitteen painikkeen painalluksen tunnistamisessa. Toinen toistettava äänitiedosto on useista eri taajuuksista muodostuva ääniaalto, chirp. Chirp-äänitiedosto toistetaan mitattavan henkilön korvaan, sen heijastuma nauhoitetaan ja lähetetään analysoitavaksi palvelimelle.

Äänen toistamista varten tarvitaan dataa, joka on tallennettu AudioBuffer-objektiin. AudioBufferiin voidaan luoda dataa decodeAudioData-funktiolla. Tämä funktio muuttaa äänitiedoston muotoon, jota voidaan käyttää äänen lähteenä. Kun data on tallennettu AudioBufferiin, luodaan AudioBufferSourceNode, jonka lähteeksi kyseinen audiodata liitetään. Luotu AudioBufferSourceNode liitetään AudioDestinationNodeen, jolloin ladattu äänitiedosto voidaan toistaa tietokoneen kaiuttimista.

3.3 Äänen nauhoittaminen

Äänen nauhoittamista varten tarvitaan WebAudio API:n tarjoamat AudioContext-, mikrofoni- ja analysaattori-objektit sekä Matt Diamondin tekemät MIT-lisenssin alaiset JavaScript-tiedostot (Diamond 2013), jotka luovat lopullisen wav-tiedoston puskuriin tallennetusta datasta.

On olemassa ohjelma (Wilson 2013), joka toimii lähes samalla tavalla kuin Otometrin nauhoitusohjelma. Ohjelma kuuntelee mikrofonia ja antaa reaaliaikaisen visuaalisen palautteen sen kuulemasta äänestä. Ohjelmalla voi myös nauhoittaa wav-äänitiedoston, jonka voi nauhoituksen jälkeen ladata koneelle.

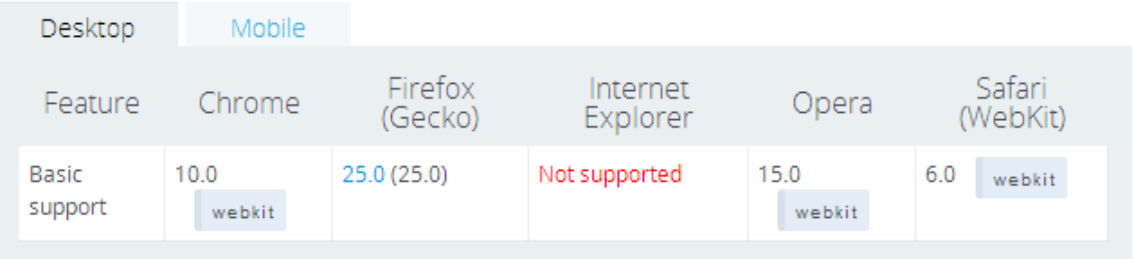
Tämä nauhoitusohjelma on periaatteessa juuri sellainen mitä tarvitsin tätä työtä varten, mutta tässä ohjelmassa äänen nauhoitus aloitetaan painamalla painiketta selaimessa, kun taas Otometrin mittausohjelmassa nauhoituksen tulisi alkaa, kun mittalaitteen painiketta painetaan. Lisäksi wav-tiedostoa ei ladata käyttäjän tietokoneelle, vaan se lähetetään Otometrin palvelimelle analysoitavaksi.

4 NAUHOITUSOHJELMAN TOTEUTUS

4.1 Äänen toistaminen

Aloitin äänen toistamiseen tutustumisen luomalla oman pienen nettisivun, jossa sivulla näkyvää painiketta painamalla toistettiin mittauksessa käytettävä chirp-äänitiedosto (kuva 8). Ennen kuin selaimessa voidaan käyttää tietokoneen äänilaitetta, pitää selaimen tukea AudioContext-elementtiä. Ohjelman tekohetkellä tuki oli Chromen 10.0:ssa ja sitä uudemmissa versioissa, Firefoxin 25.0:ssa ja sitä uudemmissa versioissa, Operan 15.0:ssa ja sitä uudemmissa versioissa, sekä Safarin 6.0:ssa ja sitä uudemmissa versioissa (kuva 9). Käytin ohjelman kehityksessä ja testauksessa Chromen versiota 35.0.

KUVA 8. HTML5-testisivusto äänen toistamiseen ja nauhoittamiseen.



Desktop | Mobile

Feature	Chrome	Firefox (Gecko)	Internet Explorer	Opera	Safari (WebKit)
Basic support	10.0 webkit	25.0 (25.0)	Not supported	15.0 webkit	6.0 webkit

KUVA 9. AudioContext-objektin tuki eri selaimissa (AudioContext 2014)

Toistettava äänitiedosto piti ladata puskuriin ennen kuin sitä käytetään. Äänitiedosto ladattiin ensin ArrayBufferiin XMLHttpRequestina, jonka jälkeen se muutettiin toistettavaan muotoon AudioContextin decodeAudioData-funktiolla (kuva 10).

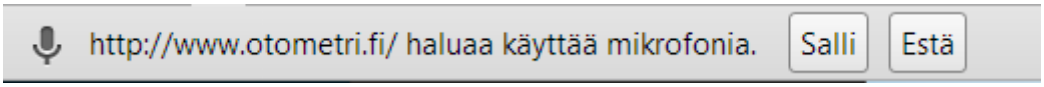
Kuvan riveillä 100–103 luodaan XMLHttpRequest, joka lataa pyydetyn tiedoston asynkronisesti. Riviltä 105 eteenpäin määritetään, mitä tehdään, kun tiedosto on ladattu. Rivillä 106 ladattu äänitiedosto muutetaan AudioBuffer-muotoon. Riveillä 107–110 ladattu äänitiedosto sijoitetaan sitä vastaavaan muuttujaan talteen myöhempää käyttöä varten.

```
96  // Funktio äänitiedoston lataamiseen valmiiksi
97  function loadSound(filename) {
98      // Debug loggausta
99      console.log('Ladataan äänitiedostoa ' + filename + ' muistiin');
100     var url = filename;
101     var request = new XMLHttpRequest();
102     request.open('GET', url, true);
103     request.responseType = 'arraybuffer';
104
105     request.onload = function() {
106         audioContext.decodeAudioData(request.response, function(buffer) {
107             if(filename=='chirp.wav')
108                 chirpBuffer = buffer;
109             else if(filename=='kohina.wav')
110                 kohinaBuffer = buffer;
111             // Debug loggausta
112             console.log('Äänitiedosto ladattu');
113         }, errorCallBack);
114     }
115     request.send();
116 }
```

KUVA 10. Äänitiedoston lataaminen AudioBufferiin

AudioContextin tarkistamisen jälkeen käyttäjältä kysytään lupa äänilaitteen käyttämiseen selaimessa. Tämä onnistui getUserMedia-funktiolla, jolle voi antaa

parametrit sen mukaan, haluaako sivulla käyttää ääntä tai videota vai molempia. Tämän funktion kutsuminen antaa ilmoituksen käyttäjälle, että selain haluaa käyttää valittua medialaitetta (kuva 11). Ilmoituksen ulkoasu vaihtelee selainten välillä.



KUVA 11. Mikrofonin käyttöoikeuksien antaminen Chrome-selaimella.

Loin funktion, jota kutsumalla luotiin AudioContext-objektin avulla uusi AudioBufferSourceNode, johon liitettiin aiemmin AudioBuffer-objektiin ladattu chirp- tai kohina-äänitiedosto. Tämän jälkeen luotu AudioBufferSourceNode liitettiin AudioDestinationNodeen, joka toisti ladatun äänen oletusäänilaitteen kaiuttimista. (Kuva 12.)

Kuvassa riveillä 131–136 luodaan uusi AudioBufferSourceNode ja siihen liitetään funktion kutsussa lähetetty AudioBuffer-objekti. AudioBufferSourceNode liitetään tämän jälkeen kaiuttimiin ja aloitetaan äänen toisto. Myös nauhoittaja alustetaan ja aloitetaan nauhoitus. Riviltä 137 eteenpäin määritellään, mitä tehdään, kun toistettava äänitiedosto loppuu. Riviltä 138 eteenpäin suoritetaan ajoitettu toiminto 200 millisekunnin kuluttua. Riveillä 141–142 pysäytetään nauhoitus ja luodaan nauhoitetusta datasta wav-äänitiedosto Matt Diamondin JavaScript-tiedostojen avulla. Rivillä 155 kutsutaan funktiota, joka lähettää äänitiedoston analysoitavaksi palvelimelle.

```
127  // Funktio joka soittaa äänen
128  function playSound(buffer) {
129      // Debug loggausta
130      //console.log('Aloitetaan äänen toisto');
131      source = audioContext.createBufferSource();
132      source.buffer = buffer;
133      source.connect(audioContext.destination);
134      recorder.init();
135      recorder.record();
136      source.start(0);
137      source.onended = function() {
138          setTimeout(function() {
139              // Debug loggausta
140              //console.log('Lopetetaan äänen nauhoitus');
141              recorder.stop();
142              recorder.exportMonoWAV(function(blob) {
143                  var url = (window.URL || window.webkitURL).createObjectURL(blob);
144                  // create a new request and send it via the objectUrl
145                  var request = new XMLHttpRequest();
146                  request.open("GET", url, true);
147                  request.responseType = "blob";
148                  request.onload = function(){
149                      // send the blob somewhere else or handle it here
150                      // use request.response
151                      var file = request.response;
152                  }
153                  request.send();
154
155                  uploadFile(blob);
156              });
157          }, 200);
158      }
159  }
```

KUVA 12. Funktio, joka toistaa AudioBufferiin ladatun äänitiedoston.

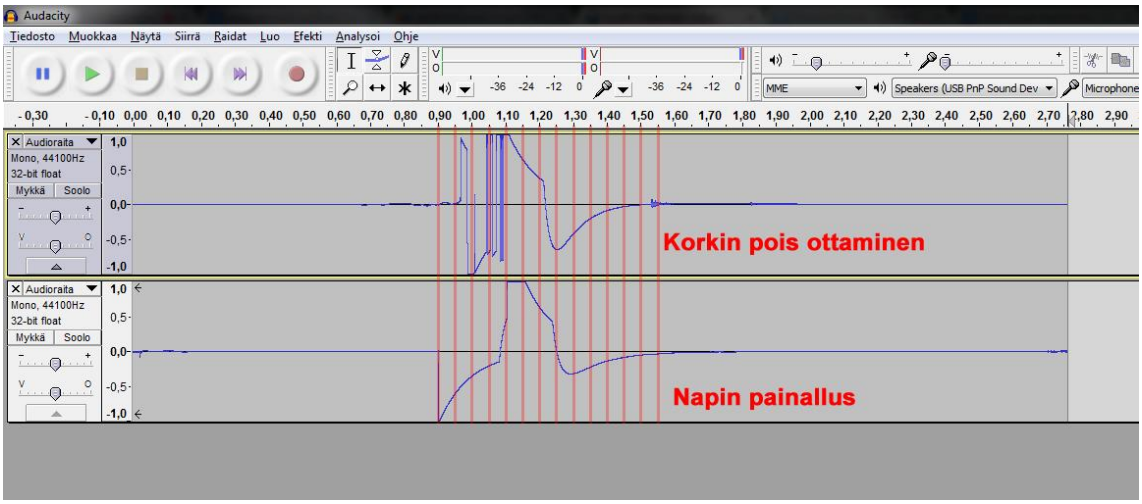
4.2 Painikkeen painalluksen tunnistaminen

Koska mittalaitteen painike on mekaaninen, ei sen tilaa voida lukea yksinkertaisilla arvoilla 0 ja 1, vaan mikrofonin pitää kuunnella, onko painike pohjassa vai ei. Ollessaan pohjassa painike aiheuttaa oikosulun, jolloin mikrofoni ei reagoi mihinkään ääneen. Mikrofonin kuuntelemaa ääntä tutkimalla voidaan päätellä, onko painike pohjassa vai ei. Aiemmin mainittu kohinaääni auttaa tämän tunnistuksessa, sillä kun epäillään, että painike on pohjassa, toistetaan mittalaitteen kaiuttimesta kohinaääntä ja samalla tarkkaillaan mikrofonin nauhoittamaa ääntä. Mikäli mikrofoni nauhoittaa edelleen hiljaisuutta, voidaan olla varmoja painikkeen pohjassa olemisesta ja valmistautua chirpin toistoon.

Yllämainittu metodi vaatii painikkeen pohjassa pitämistä hetken aikaa, jonka jälkeen se vapautuessaan toistaa chirpin. Yhteensä tämä tapa vie aikaa noin 3

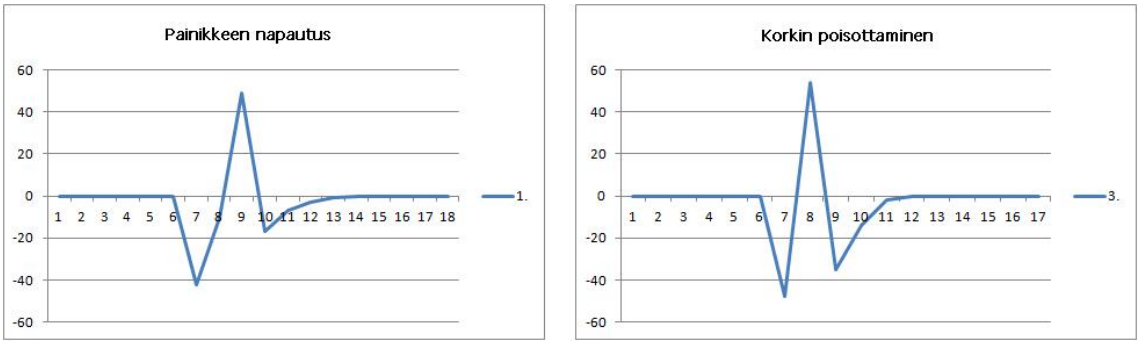
sekuntia per mittaus. Pyrin tämän työn aikana jättämään kohinaäänen pois ja luomaan painikkeen tunnistuksen, joka toimisi pelkällä painikkeen napauttamisella.

Tutkin painikkeen painalluksesta muodostuvaa ääniaaltoa (kuva 13) ja pyrin löytämään siitä yksilöivän muodon, joka toistuisi vain ja ainoastaan, kun painiketta napautetaan. Pääpirteiltään mittalaitteen painikkeen napauttaminen ja korkin poisottaminen kuitenkin tuottavat samannäköisen ääniaallon. Korkin poisottaminen aiheuttaa keskellä ääniaaltoa hyvin voimakasta liikettä laidasta toiseen, mutta koska se tapahtuu erittäin lyhyen ajan sisään, sitä ei voida hyödyntää painikkeen painalluksen tunnistamisessa. Mittausohjelmassa mikrofonin nauhoittama data tutkitaan 50 millisekunnin välein (punaiset pystyviivat kuvassa 13), jolloin korkin poisottamisen aiheuttama voimakas liike jää usein huomaamatta ja tästä syystä näyttää samanlaiselta kuin painikkeen painallus.



KUVA 13. Mittalaitteen painikkeen napauttamisen ja korkin poisottamisen erot.

Tein myös testejä, joissa nauhoitin uudella HTML5-mittausohjelmalla painikkeen napauttamista ja korkin poisottamista, jolloin sain 50 millisekunnin välein arvoja, jotka syötin Exceliin ja loin niiden pohjalta kuvaajia. Kuvaajista osoittautuu selkeästi, että mittausohjelmisto näkee mittalaitteen painikkeen napautuksen ja korkin poisottamisen samanlaisena. (Kuva 14.)

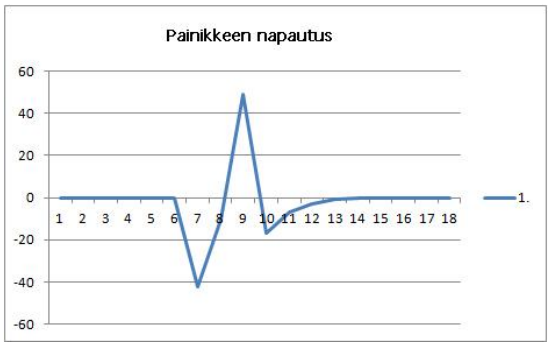


KUVA 14. Painikkeen napautuksen ja korkin poisottamiseen yhdennäköisyys.

Edellä olevien testien perusteella tulin siihen lopputulokseen, että nykyisellä mittauslaitteistolla ei ole mahdollista päästä eroon kohinaäänestä ja pitää yllä 100-prosenttista painikkeen painalluksen tunnistusta. Niinpä sovitin Java-ohjelmassakin käytetyn painikkeen painamisen tunnistamiseen käytetyn algoritmin HTML5-versioon.

4.3 Äänen nauhoittaminen

Äänen tutkimista varten mikrofoni-objektiin piti liittää AnalyserNode-objekti (Adenot – Wilson 2013d), joka ottaa mikrofonin nauhoittaman datan vastaan ja lähettää sen eteenpäin muuttumattomana, mutta samalla tuottaa reaaliaikaista taajuus- ja aikapohjaista datainformaatiota sille syötetystä äänestä. Käytin mittausohjelmassa AnalyserNoden getByteTimeDomainData-funktiota, joka palauttaa viimeisimmän halutun määrän näytteitä arvovälillä 0–256. Laskemalla saaduista arvoista keskiarvo saadaan yksittäinen piste ääniaallossa. Kun näitä pisteitä luodaan useampia kappaleita 50 millisekunnin välein, saadaan muodostettua ääniaalto. (Kuva 15.)



KUVA 15. Ääniaalto, joka on muodostettu 50 millisekunnin välein otetuista näytteistä.

Tutkimalla 50 millisekunnin välein saatuja arvoja löysin raja-arvot, joiden sisällä arvot olivat, kun painiketta pidetään pohjassa. Kun arvot olivat raja-arvojen sisällä, laitoin kohinaäänen toistumaan mittalaitteen kaiuttimesta ja seurasin edelleen mikrofonin nauhoittamaa ääntä. Mikäli arvot pysyivät edelleen raja-arvojen sisällä, painike oli pohjassa. Seuraavan kerran, kun arvot menivät raja-arvojen ulkopuolelle, oli painike päästetty pohjasta ja chirp toistettiin ja nauhoitettiin sekä lähetettiin palvelimelle käyttämällä aikaisemmin kuvattua playSound-funktiota.

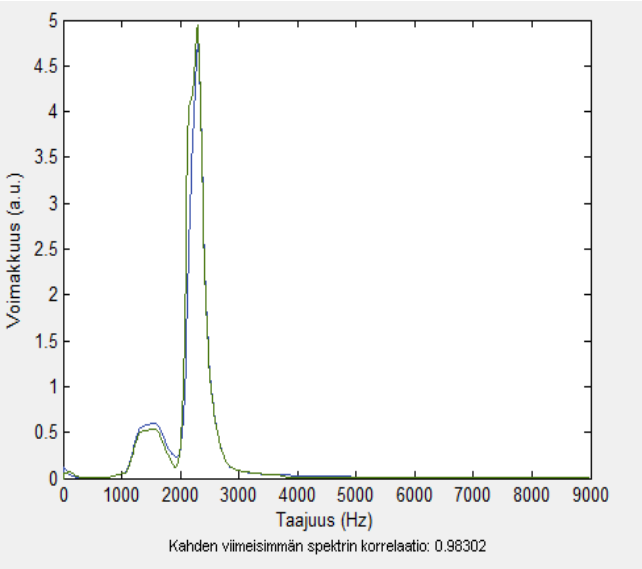
4.4 Ääniasetusten säätäminen

Java-versiossa oli mahdollisuus päästä käsiksi käyttöjärjestelmän käyttämiin äänilaitteisiin ja muokata äänenvoimakkuuden tasoja ja mikrofonin ja kaiuttimen mykistyksiä automaattisesti samalla kun mittausohjelma käynnistyi. Koska HTML5 ja selain toimivat korkeammalla ohjelmistotasolla, kuin Java-ohjelmat, ei siitä ole pääsyä suoraan äänilaitteiden asetuksiin. Näin ollen HTML5-versiossa pitää tukeutua käyttäjän opastamiseen siitä, että ääniasetukset ovat asetettuna oikein ennen mittauksen aloittamista. Asiaa helpottaa se, että käyttöjärjestelmä muistaa äänilaitteelle asetetut ääniasetukset, kunhan käytetään samaa USB-porttia eri kertojen välillä.

5 TULOSTEN VERTAILU

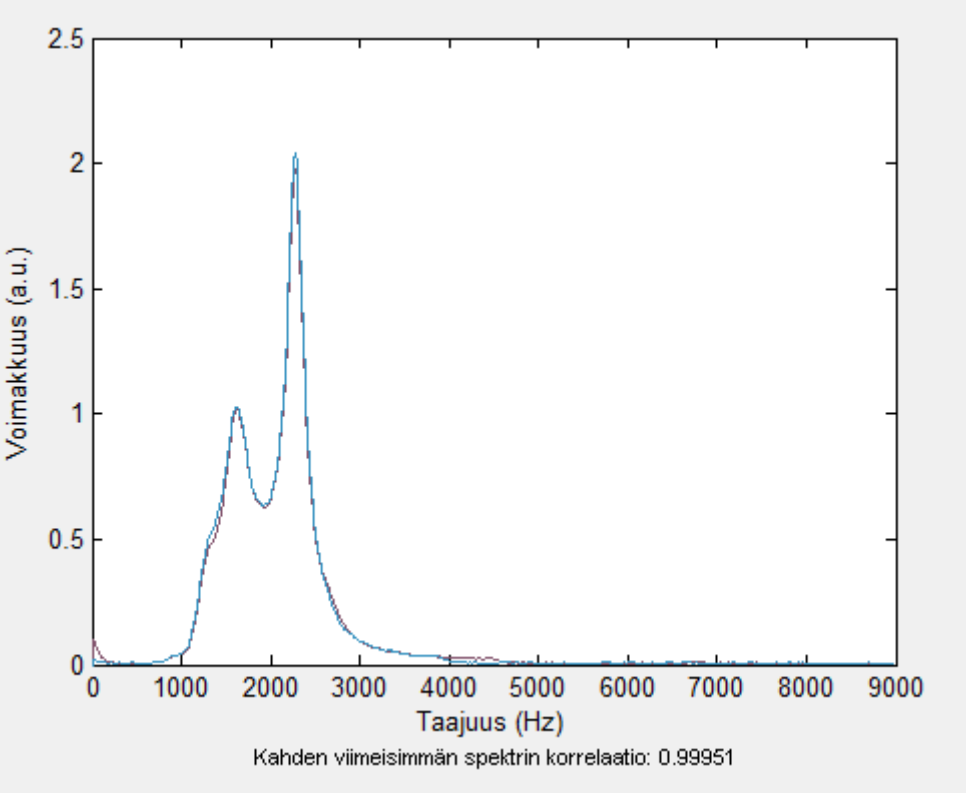
Otometrin mittalaite on kliinisesti tutkittu mittalaite (Otometri. 2009), joten uuden HTML5-version tuli tuottaa samanlaisia wav-tiedostoja ja vastaavia tuloksia, kuin mitä Java-versio tuotti. Tätä tutkimusta varten työn tilaaja loi minulle käyttöön testiohjelman, jolla pystyin tutkimaan sille syötetyn ääniaallon spektriä ja vertaamaan kahden viimeisimmän spektrin korrelaatiota. Mikäli kahden spektrin korrelaatio on 1, spektrit ovat täysin samanlaiset, mikä tarkoittaa sitä, että HTML5-versiolla saadut tulokset ovat yhtä luotettavia, kuin Java-versiolla saadut.

Mittauksissa käytin Otto II:ksi nimettyä testilaitetta, jolla on tervettä ja tulehtunutta korvaa vastaavat keinokorvat. Tein kymmenen testimittausta molemmilla ohjelmistoversioilla, Javalla ja HTML5:llä, sekä tulehtuneesta että terveestä korvasta. Vertasin mittauksissa nauhoitettuja äänitiedostoja testiohjelman avulla syöttämällä ohjelmalle ensin yhden Java-version nauhoittaman äänitiedoston ja tämän jälkeen HTML5-version nauhoittaman vastaavan äänitiedoston. Testiohjelma antoi kaikista äänitiedostopareista oman korrelaatioarvon ja piirsi äänitiedostojen spektrit (kuva 16).



KUVA 16. Java- ja HTML5-versioiden tulehtuneen keinokorvamittauksen spektrit.

Kuvista ilmenee myös, kuinka tulehtunut ja terve korva antavat erilaisen spektrin (kuva 17).



KUVA 17. Java- ja HTML5-versioiden terveen keinokorvamittauksen spektrit.

Testeistä saadut korrelaatioarvot olivat välillä 0,98302–0,99988. Testitulosten pohjalta voitiin todeta, että HTML5-nauhoitusohjelman tulokset vastaavat Java-ohjelman tuloksia ja HTML5-ohjelma on käyttökelpoinen.

6 YHTEENVETO

Työn tarkoituksena oli kehittää Otometri-mittalaitteelle uusi HTML5-mittausohjelmisto nykyisen Java-ohjelman rinnalle tai tilalle. Tällä uudella tekniikalla luodulla mittausohjelmalla käyttäjä säästyisi mittausohjelman erillisestä lataamisesta omalle koneelle ja mittaus tapahtuisi kokonaan selaimessa. Myöskään Javan ei tarvitsisi olla asennettuna käyttäjän koneella, mikä osaltaan lisää käyttäjän omaa tietoturvaa. HTML5-ohjelmalla tehty mittausohjelma antaa mahdollisuuden tulevaisuudessa tehdä mittauksia myös mobiililaitteilla.

HTML5-mittausohjelman tekeminen onnistui hyvin ja vaaditulla tasolla. Vaikka HTML5-versiossa ääniasetuksia ei ole mahdollista asettaa automaattisesti oikealle tasolle, tuo se mukanaan muita mittausta parantavia ominaisuuksia, minkä perusteella työ katsottiin onnistuneeksi ja otettiin käyttöön ensisijaiseksi mittausohjelmaksi. Vanhan Java-version käyttömahdollisuus säilytettiin ja käyttäjä voi halutessaan valita ennen mittausta kumpaa mittausohjelmaa käyttää, HTML5:tä vai Javaa.

Koska HTML5 on vielä kehitteillä, ei sen tarjoamia ominaisuuksia ole vielä lisätty kaikkiin yleisimpiin saatavilla oleviin selaimiin. HTML5-versiota ei voinut käyttää Internet Explorerilla ollenkaan, mutta yllämainitulla tavalla vaihtoehtoiseksi mittausohjelmaksi jätettiin myös aikaisempi Java-versio vastaavanlaisia tapauksia varten.

LÄHTEET

Adenot, Paul – Wilson, Chris 2013a. AudioBufferSourceNode. Saatavissa: http://www.w3.org/TR/webaudio/#AudioBufferSourceNode. Hakupäivä 22.8.2014.

Adenot, Paul – Wilson, Chris 2013b. AudioContext Interface. Saatavissa: http://www.w3.org/TR/webaudio/#AudioContext-section. Hakupäivä 22.8.2014.

Adenot, Paul – Wilson, Chris 2013c. AudioDestinationNode. Saatavissa: http://www.w3.org/TR/webaudio/#AudioDestinationNode. Hakupäivä 22.8.2014.

Adenot, Paul – Wilson, Chris 2013d. AnalyserNode. Saatavissa: http://www.w3.org/TR/webaudio/#AnalyserNode. Hakupäivä 22.8.2014.

Akustinen reflektometria ja neuroverkot Otometrin ytimenä. 2009. Otometri. Saatavissa: http://www.otometri.fi/images/akustinen%20reflektometria%20ja%20neuroverkot%20otometrin%20ytimen.pdf. Hakupäivä 22.8.2014

AudioContext. 2014. Mozilla Developer Network. Saatavissa: https://developer.mozilla.org/en-US/docs/Web/API/AudioContext. Hakupäivä 11.9.2014.

Diamond, Matt 2013. Recorderjs. Saatavissa: https://github.com/mattdiamond/Recorderjs. Hakupäivä 22.8.2014.

Hoida korvatulehdusta jo kotona. 2014. Otometri Saatavilla: http://www.otometri.fi/. Hakupäivä 26.9.2014.

Java-estot parantavat tietoturvaa, toimi näin. 2014. Viestintävirasto. Saatavissa: https://www.viestintavirasto.fi/tietoturva/tietoturvanyt/2014/07/ttn201407091103.html. Hakupäivä 22.8.2014.

NavigatorUserMedia.getUserMedia(). 2014. Mozilla Developer Network. Saatavissa: https://developer.mozilla.org/en-US/docs/NavigatorUserMedia.getUserMedia. Hakupäivä 11.9.2014.

Mills, Chris 2014. Web Audio API. Saatavissa: https://developer.mozilla.org/en-US/docs/Web/API/Web_Audio_API. Hakupäivä 11.9.2014.

Otometri. 2009. Otometri Saatavilla: http://www.otometri.fi/fi/korvatulehdusmittari.html. Hakupäivä 26.9.2014.

Web Audio API. 2014. Mozilla Developer Network. Saatavissa: https://developer.mozilla.org/en-US/docs/Web/API/Web_Audio_API. Hakupäivä 11.9.2014.

Web Audio API support. 2014. Microsoft Connect. Saatavissa: https://connect.microsoft.com/IE/feedback/details/799529/web-audio-api-support. Hakupäivä 26.9.2014.

Wilson, Chris 2013. AudioRecorder. Saatavissa: http://webaudiodemos.appspot.com/AudioRecorder/index.html. Hakupäivä 22.8.2014.

Ääni. 2014. Wikipedia. Saatavissa: http://fi.wikipedia.org/wiki/Ääni Hakupäivä 11.9.2014.