Janne Martikainen

PHP-pohjainen hallintatyökalu verkkopalvelulle

Metropolia Ammattikorkeakoulu

Insinööri (AMK)

Tietotekniikka

Insinöörityö

14.4.2014

Metropolia

Tekijä(t) Otsikko Sivumäärä Aika	Janne Martikainen PHP-pohjainen hallintatyökalu verkkopalvelulle 40 sivua 14.4.2014
Tutkinto	Insinööri (AMK)
Koulutusohjelma	Tietotekniikan koulutusohjelma
Suuntautumisvaihtoehto	Ohjelmistotekniikka
Ohjaaja(t)	Lehtori Ilpo Kuivanen talousjohtaja, Toinen veli Oy, Valter Pasanen

Tämä työ tehtiin Toinen veli Oy:lle. Toinen veli Oy on kehittänyt kiinteistöhuoltoyrityksille suunnattua toiminnanohjausjärjestelmää nimeltä KiinteistöVELI. Sovellusta myydään asiakkaille palveluna.

Tämän työn tavoitteena oli kehittää hallintasovellus, joka tuottaa käyttöliittymän käyttötapauksille, kuten asiakkaan lisääminen palveluun. Toteutuksen tekniikoina käytettiin PHP:tä ja standardin mukaisia selaintekniikoita. Työn toisena tarkoituksena oli olla prototyyppi muutamalle uudelle tekniikalle. Yhtiöstä haluttin, että työssä käytetään Codeigniter MVC -frameworkia, jonka käyttöä yhtiössä harkittiin tulevaisuudessa muidenkin sovellustuotteiden osalta. Työ tarkasteli myös mahdollisuuksia käyttää HTML5:ä, joka on edelleen keskeneräinen standardi.

Tässä kirjallisessa osuudessa tarkastelen käytettyjä tekniikoita yleisestä näkökulmasta. Tarkastelen myös selviä hyötyjä, jotka mielestäni saavutettiin käyttämällä työssä Codeigniterin MVC-mallia ja HTML5:tä.

Avainsanat	Codeigniter, PHP, MVC, HTML5, Verkkosovellukset

Metropolia

Author(s) Title Number of Pages Date	Janne Martikainen PHP-based administration tool for online service 40 pages 4 April 2014
Degree	Bachelor of Engineering
Degree Programme	Information technology
Specialisation option	Software engineering
Instructor(s)	Ilpo Kuivanen, Senior lecturer Valter Pasanen, Financial director, Toinen veli Ltd.

The study was creted for Toinen veli Ltd. The company has developed an enterprise resource planning (ERP) software called KiinteistöVELI directed to housing maintenance companies. The software is sold as a service to the client companies.

The purpose of the study was to develop an administration tool that would implement a user interface for tasks such as adding customers to the service. The technologies used for the implementation were PHP and standardized web browser technologies. Another purpose of the project was to serve as a prototype for a few new technologies. The company wanted the project to use the Codeigniter MVC framework that was considered to be used with other software products in the future. The study also looked into using HTML5 which is still an unfinished standard.

The study discusses about the web technologies from a general viewpoint as well as the actual benefits achieved by using HTML5 and codeigniter's MVC model.

Keywords	Codeigniter, PHP, MVC, HTML5, Web application

Metropolia

Sisällys

Sanasto

AJAX	Lyhenne sanoista Asynchronous Javascript and XML. tekniikka, jolla on mahdollista tehdä dynaamisia verkkopyyntöjä Javascript-koodilla.
API	Application Programming Interface tai ohjelmointirajapinta. Dokumentaatio, joka kuvaa sovellusrajapinnan osat, sekä miten niitä käytetään.
ASP	Active Server Pages. Microsoftin palvelinohjelmointitekniikka.
Client-side	Asiakaspuoli. Viittaa verkkosovelluksessa käyttäjään ja käyttäjän selaimeen, jossa suoritetaan palvelimelta saatua HTML:stä, CSS:stä ja Javascriptista koostuvaa verkkosivustoa.
CSS	Cascading Style Sheets. Merkintätapa, jolla määritellään tyyliasetuksia eri käyttöliittymäosille. Usein omana erillisenä dokumenttinaan, jonka tiedostotyyppinä on .css.
DOM	Document Object Model. Määritys miten HTML- tai XML-dokumentin tuottamaa puurakennetta voidaan ohjelmallisesti navigoida tai muokata. Mahdollistaa esim. HTML-elementtien lisäämisen Javascriptista.
ECMAScript	Skriptikieli. Yleisin käyttö selainohjelmoinnissa, jossa yhteydessä tämä kieli tunnetaan paremmin nimellä Javascript.
Flash	Adoben kehittämä sovellusalusta ja selainlisäosa, jolla voidaan toistaa videota ja muuta multimediasisältöä.
HTML	Hypertext Markup language. Verkkosivujen luomiseen käytettävä merkintäkieli.
HTTP	Hypertext transfer protocol. Tiedonsiirtoprotokolla, jonka ovat kehittäneet W3C ja Internet Engineering Task Force.

Javascript	kts. ECMAScript.
JQuery	Suosittu Javascript-kirjasto. Luultavasti tunnetuin ominaisuudestaan poimia HTML-elementtejä CSS-valitsimilla.
JSON	Javascript Object Notation. Tiedonsiirtomuoto, jolla voidaan esim. sarjallistaa Javascript-olioita.
LAMP	Lyhenne sanoista Linux, Apache, MySQL, ja PHP. Viittaa ohjelmistopinoon, joka muodostaa ns. kolmikerrosarkkitehtuurin.
MVC	Model-View-Controller. Ohjelmistotuotantomalli, joka on alunperin tarkoitettu selkeyttämään graafisten työpöytäsovellusten koodin muokattavuutta ja luettavuutta.
PHP	Hypertext Preprocessor. Ohjelmointikieli, jonka yleisin käyttö on palvelinohjelmoinnissa.
Polyfill	Verkkomaailmassa käytettävä termi, jolla viitataan yleensä Javascript-kirjastoon, joka mahdollistaa uuden ominaisuuden käyttämistä verkkoselaimessa, joka ei ominaisuutta luontaisesti tue. voi olla välikappale, joka sovittaa standardin mukaisen koodin selainkohtaiseen ominaisuuteen tai hyödyntää jotain alemman tason ominaisuutta toteuttaakseen uutta API:a.
SGML	Standard Generalized Markup Language. W3C:n määrittelemä merkintäkieli, jonka alikieliä ovat XML sekä HTML 4.01 ja sitä aikaisemmat HTML-versiot.
SQL	Standard Query Language. Alkujaan IBM:n kehittämä tietokantojen kyselykieli, jota käytetään relaatiotietokannan tietojen hakemiseen ja muokkaamiseen.
W3C	World Wide Web Consortium. Standardointiorganisaatio, joka standardoi verkkotekniikoita. Ylläpitää mm. HTML- , XML- ja HTTP-standardeja.

WHATWG Web Hypertext Application Technology Working Group. Applen, Mozillan ja Operan alulle laittama työryhmä, joka luo erilaisia verkkotekniikoiden määrityksiä. Tuottanut mm. pohjan viralliselle HTML5-määritykselle.

XHTML XML-merkintäkielellä kirjoitettu verkkosivusto. Syntaksisäännöiltään tiukempi kuin HTML.

XML Extensible Markup Language. W3C:n määrittelemä yleiskäyttöinen merkintäkieli. Toimii pohjana useille muille merkintäkielille esim. MathML ja SVG.

1 Johdanto

Tämä työ tehtiin Toinen veli Oy:lle. Yritys on tuottanut KiinteistöVELI-nimistä verkkosovellusta, joka on kiinteistöhuoltoyrityksille suunnattu toiminnanohjaus-järjestelmä. Sovellusta myydään yrityksille palveluna, jossa jokaisella yrityksellä on oma tietovarastonsa, johon asiakasyhtiöt voivat tallentaa kiinteistöihin liittyviä huoltotapahtumia, autopaikkavarauksia sekä muita kiinteistöihin liittyviä tietoja. Järjestelmä mahdollistaa myös, että huoltoyritykset voivat vastaanottaa sähköisesti esim. saunavuorovarauksia. Palvelusta laskutetaan kiinteistöjen määrän mukaan ja asiakkaalle on asetettu järjestelmään laskutuksen mukaan sovittu taloyhtiöraja, jota kiinteistöhuoltoyritys ei voi ylittää. Tämän työn tavoitteena oli tuottaa Toinen veli Oy:n sisäiseen käyttöön tuleva hallintapaneeli, jolla olisi mahdollista suorittaa erinäisiä hallitatoimenpiteitä KiinteistöVELI-sovellukseen.

Hallintapaneelin oli tarkoitus olla selaimessa toimiva sovellus, joka tarjoaisi yksinkertaisen käyttöliittymän monille hallintatoimenpiteille, joiden suorittamiselle ei ennen tätä sovellusta ollut erillistä omaa työkaluaan. Hallintapaneelin kautta piti olla mahdollista lisätä palveluun uusia asiakkaita ja hallita asiakkaaseen liittyviä tietoja. Hallintapaneelin ei ollut tarkoitus hallinnoida asiakkaan laskutusta eikä muita vastaavia tietoja. Hallintapaneelilla piti myös olla mahdollista myös muokata asiakkaiden tietoja kuten taloyhtiörajaa sekä asiakkaalle kuuluvia käyttäjätunnuksia, jolla asiakkaan tietovarastoon voitaisiin kirjautua. Sovelluksen piti olla myös mahdolllista näyttää lokitiedostoja, joita palvelu oli tuottanut. Hallintapaneeliin toteutettaisiin oma kirjautumisikkunansa, ja sovellus käyttäisi omaa käyttäjätietokantaansa kirjautumis-tietojen todentamiseen.

Työn palvelinsovellus toteutettiin PHP:llä ja asiakaspuolella käytettiin standardin mukaisia selaintekniikoita eli CSS:ää, Javascriptia ja HTML:ää. Työn oli tarkoituksena toimia myös eräänlaisena prototyyppinä muutamille uusille tekniikoille. Palvelinsovelluksessa hyödynnettiin Codeigniter-frameworkia, johon siirtymistä yhtiössä harkittiin muidenkin sovellustuotteiden osalta. Tarkoituksena oli myös tarkastella mahdollisuuksia hyödyntää HTML5:tä ja sen tarjoamia ominaisuuksia. HTML5:n uusia ominaisuuksiakaan ei ollut tarkoitus änkeä väkisin työhön vaan ainoastaan parantamaan sovellusta, jos niiden käyttömahdollisuudet olisivat realistisia myös selainyhteensopivuuden kannalta. Työn tekemisessä ei käytetty mitään

varsinaista kehitysympäristöä vaan työkaluna toimi pääasiassa Linux-terminaalissa toimiva VIM-tekstieditori(VI Improved). Selainpuolella käytettiin asemoinnin tarkasteluun ja Javascript-koodin debuggaamiseen Firebug-lisäosaa, joka on saatavilla mm. Google Chrome ja Mozilla Firefox selaimiin. Javascriptissa käytettiin Jquery- ja JQuery UI -kirjastoja.

Työtä tehtiin pääasiassa etätyönä, mutta sovellustuotannon tiimoilta käytiin useampia tapaamisia, joissa käytiin aina läpi aikaisemmassa tapaamisessa sovitun sovellus-kokonaisuuden toteutus sekä sovittiin seuraavan aikajakson aikana toteutettava osa-alue sekä mahdollisesti aikaisemmasta toteutuksesta esiin nousseet korjaustarpeet. Sovelluksella oli tietyt vaatimukset tietoturvan ja käytettävyyden osalta, mutta muutoin sovelluksen toteutus oli melko vapaata. Sovelluksen piti olla helppokäyttöisyydeltään sitä luokkaa, ettei sovelluksen käyttäminen vaatinut mitään suurta teknistä osaamista. Selainyhteensopivuuden puolelta toivottiin, että sovellus olisi käytettävissä IE8-tasoisilla selaimilla. Ulkoasun suhteen pyrittiin samankaltaiseen ulkoasuun, kuin mitä KiinteistöVELI sillä ajanhetkellä käytti.

2 Verkkosovellukset

Nykyaikaisen verkkosivuston ja verkkoselainten takana on iso joukkko standardeja, joilla pyritään takaamaan laaja yhteensopivuus lukuisille eri laitteistoille ja järjestelmille. Suurinta osaa näistä standardeista ylläpitää World Wide Web Consortium tai lyhyemmin W3C. Verkkosivustot, joita selaimella tarkastellaan, tuotetaan pääosin kolmella standardisoidulla selaintekniikalla, joita ovat HTML, CSS ja Javascript tai virallisemmin ECMAScript. Selain kommunikoi verkkopalvelinten kanssa HTTP-protokollaa käyttäen. Verkkopalvelin vastaanottaa selainten tekemät pyynnöt ja vastaa niihin. Palvelimen vastaus voi olla dataa, tai vain vastaus, että pyydetty resurssi ei ole ko. palvelimelle. Yksinkertaisimmissa tapauksissa verkkopalvelimella on vain HTML-tiedostoja, jotka palvelin lähettää selaimille kun tiedostoa pyydetään. Verkkopalvelinten on myös mahdollista vastaanottaa dataa käyttäjältä ja palvelin voidaan ohjelmoida tuottamaan sivu dynaamisesti ja käsittelemään käyttäjän lähettämä data. Verkko-palvelimen ohjelmointiin on useita tekniikoita, kuten esim. PHP, ASP ja Java EE. Ohjelmoitavan verkkopalvelimen kanssa käytetään yleensä myös relaatiotietokantaa, joka mahdollistaa tietojen tallentamisen ja nopean hakemisen. Relaatiotietokanta voi olla fyysisesti samassa laitteistossa kuin verkkopalvelinkin tai erillään siitä. Verkkopalvelin tekee tietokantaan kyselyitä, joilla pyydetään tietyn tyyppisiä tietueita tai

muokataan määritettyjä tietueita. Yksi yleisimmistä relaatiotietokantojen kyselykielistä on SQL (Standard Query Language). Edellä mainitut kolme osaa: Selain, Verkkopalvelin ja relaatiotietokanta muodostavat kokonaisuuden, jota kutsutaan kolmikerros-arkkitehtuuriksi. Yksi tunnetuimmista ohjelmistopinoista, joka muodostaa tällaisen kolmikerrosarkkitehtuurin on LAMP, joka on lyhenne sanoista Linux, Apache, MySQL ja PHP.

2.1 HTML-merkintäkieli

HTML-merkintäkieli kuvaa verkkosivun rakenteen. HTML on merkintää, jota ihminenkin pystyy lukemaan. Seuraavassa koodiesimerkissä on lyhyt pätkä HTML-merkintää.

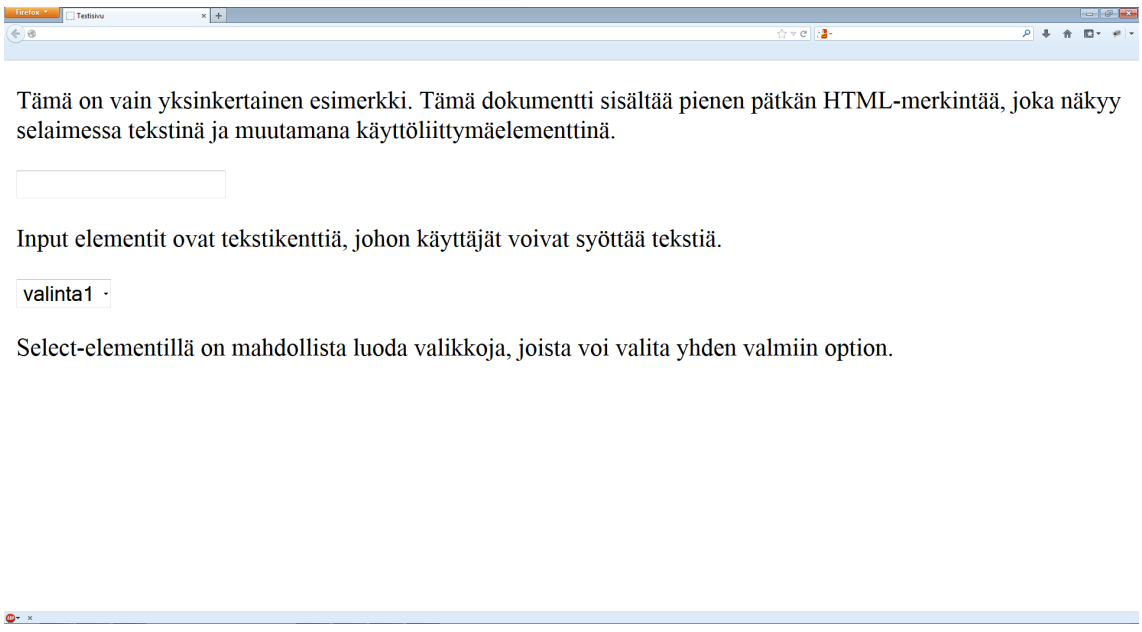
```
<!DOCTYPE html>
<html>
    <head>
        <title>Testisivu</title>
        <meta http-equiv="Content-Type"
           content="text/html;charset=utf-8" >

   </head>
   <body>
       <p class="paragtext" >
           Tämä on vain yksinkertainen esimerkki. Tämä dokumentti
           sisältää pienen pätkän HTML-merkintää, joka näkyy
           selaimessa tekstinä ja
           muutamana käyttöliittymäelementtinä.
       </p>
       <input type="text" id="input1" class="datainput" />
       <p class="paragtext" >
           Input elementit ovat tekstikenttiä, johon käyttäjät
           voivat syöttää tekstiä.
       </p>
       <select id="select1" class="datainput" >
           <option value="1">valinta1</option>
           <option value="2">valinta2</option>
       </select>
       <p class="paragtext">
           Select-elementillä on mahdollista luoda valikkoja,
           joista voi valita yhden valmiin option.
       </p>
   </body>
</html>
```

Koodiesimerkki 1. Yksinkertainen HTML-dokumentti

Esimerkki on tehty HTML5:llä, minkä takia alussa on vain yksinkertainen doctype-määritys, jonka ainoana tarkoituksena on taata että dokumentti toimii myös vanhemmilla selaimilla. HTML-merkintä koostuu elementeistä, jotka voivat sisältää

toisia elementtejä, muodostaen näin puurakenteen. Elementillä on aina aloitus- ja lopetusmerkintänsä, joita kutsutaan HTML-tageiksi. Aloitus- ja lopetusmerkintöjen välissä olevat merkinnät ovat elementin sisältö. Aloitusmerkintä on aina muotoa <elementin-nimi> ja lopetusmerkintä on muodoltaan </elementin-nimi>. Elementit voidaan myös aloittaa ja päättää lyhyemmällä muodolla <elementin-nimi /> silloin kun elementillä ei ole mitään sisältöä. . HTML-merkinnässä on XML:n tavoin yksi juurielementti, joka HTML-dokumentin tapauksessa on aina html-elementti. Elementeillä voi olla myös attribuutteja, jotka asetetaan aloitusmerkintään heti elementin nimen jälkeen. Esimerkistä löytyy attribuutti mm. input-elementistä, jolla on type-, id- ja class-attribuutit. id- ja class-attribuutit on mahdollista laittaa lähes jokaiselle elementille, ja niitä käytetään yleensä CSS-tyylimääritysten kohdistamiseen. HTML-dokumentti alkaa aina DOCTYPE-määrityksellä, joka ennen HTML5:ä määritti mitä HTML-versiota dokumentti käytti. HTML5:ssä DOCTYPE on olemassa enää vain taaksepäin yhteensopivuuden takaamiseksi. HTML-dokumentilla on aina juurielementti html, jonka sisällä ovat kaikki muut HTML-elementit. Seuraavassa kuvassa näkyy miltä esimerkin HTML näyttää selaimessa.



Kuva 1. HTML-esimerkki Firefox-selaimessa

Selain vastaa HTML-dokumentin visualisoinnista. Eri selainten välillä voi olla pieniä eroja käyttöliittymien ulkoasussa, mutta jos selain toteuttaa standardeja oikein ei selaimen tuottama esitys pitäisi olla merkittävästi erilainen toiseen selaimeen verrattuna. Tavanomaisen HTML:n lisäksi verkkosivuja on mahdollista tuottaa XML-

merkintäkielellä. XML esitystapa tunnetaan nimellä XHTML. XML:n syntaksi on samanlainen kuin HTML:n, mutta XML on oikeinkirjoitussäännöiltään tiukempi kuin HTML. HTML:ssä on mahdollista jättää joitain elementtejä päättämättä, kun taas XHTML vaatii että kaikki elementit pitää ehdottomasti päättää ja dokumenttiin pitää lisätä XML:n nimiavaruusmerkinnät.

2.2 Javascript

Javascript, viralliselta nimeltään ECMAScript, on ohjelmointikieli, jolla voidaan kirjoittaa HTML-sivulle sovelluskoodia. Javascript on skriptikieli, mikä tarkoittaa sitä, että Javascriptia ei käännetä palvelimen puolella kone- tai tavukoodiksi vaan lähetetään sellaisenaan käyttäjän selaimeen, joka kääntää tai tulkitsee javascript-koodin, minkä vuoksi sovelluskoodia voivat lukea helposti myös sivuston käyttäjät. Javascript-koodin lukemista voi vaikeuttaa sotkemalla (obfuscate) koodia niin että koodista poistetaan kaikki tabulaattorit ja rivivaihdot ja kaikki muuttujat ja metodit nimetään lyhyesti niin, ettei nimestä selviä varsinainen toiminta. Positiivisena lisänä tällaisessa koodissa on se, että Javascript-dokumentti pienenee kun koodista poistetaan javascript-moottorille turhat merkit (rivinvaihdot ja tabulaattorit). Javascript-koodia ei kuitenkaan voi mitenkään täysin piilottaa sivuston käyttäjältä. Verkkokehittäjän on syytä aina muistaa että Javascriptin suorittaminen on täysin asiakaspuolella tehtävä toimenpide ja käyttäjä voi jopa ottaa Javascriptin suorittamisen kokonaan pois päältä selaimestaan. Javascriptia runsaasti käyttävän sivuston olisi syytä näyttää käyttäjälle virhemerkintä, jos käyttäjällä ei ole Javascriptia käytässä. Standardisointijärjestö ECMA vastaa varsinaisen ohjelmointikielen määrityksistä, mutta verkkosovellus API:n, määrittelee W3C eli selain ympäristössä ajettavan Javascriptin yleiset funktiot ja luokat perustuvat näihin määrityksiin. Javascriptia ei alunperin ole käytetty laajasti johtuen osittain siitä että koodi ei sen dynaamisen luonteensa vuoksi ole ollut kovin tehokasta, mutta tietokonetehojen kasvun myötä ja Javascript-moottoreiden kehittymisen vuoksi Javascriptia käytetään todella paljon nykyaikaisissa verkkosovelluksissa. HTML5 laajentaa olemassaolevaa verkkosovellus API:a. Alla on yksinkertainen esimerkki javascriptista.

```
window.onload = function() {
    /* kommentti. Tämä on funktio,
    joka suoritetaan kun selain
    on ladannut sivun
    */
    alert(”hello world”);
};
```

Koodiesimerkki 2. Yksinkertainen Javascript-esimerkki.

Esimerkin koodi saa selainympäristössä selaimen antamaan erillisen dialogin, jossa lukee teksti 'hello world'. Javascriptia voidaan liittää HTML:ään script-elementillä joko niin että sovelluskoodikin on osa HTML-dokumenttia tai script-elementissä voidaan määrittää src-attribuutilla palvelimelle sijaitseva erillinen tiedosto, joka sisältää javascriptkoodin. Erillinen tiedosto on usein suositeltavampi vaihtoehto.

DOM eli Document Object Model on sovellusrajapinta, jolla voidaan ohjelmallisesti käsitellä dokumentteja, jotka ovat HTML- tai XML-muodossa. DOM mahdollistaa HTML:n käsittelemisen Javascriptilla. DOM määrittää useita metodeja, joilla voidaan navigoida merkintöjen muodostamaa puurakennetta ja käsitellä olemassaolevia elementtejä esim. lisäämällä uusia elementtejä olemassaolevan elementin alle. DOM:lla on useita versioita, jotka tunnetaan yleensä DOM:n tasoina. Tällä hetkellä W3C on työstämässä DOM level 4 -määritystä.

AJAX eli Asynchronous Javascript and XML on yleisnimitys tekniikkalle, jossa Javascript-koodissa voidaan tehdä taustalla (asynkronisesti) pyyntöjä palvelimelle ja käsitellä palvelimen toimittama vastaus, sekä mahdollisesti käsitellä avonaista sivua DOM:in avulla. AJAX mahdollistaa monissa tilanteissa perinteistä verkkosivua tehokkaakkaamman toiminnan, sillä selain voi vastaanottaa vain pienen osan sisältöä, joka lisätään auki olevan sivun sisälle. Tekniikan nimestä huolimatta vastauksena saadun datan ei tarvitse olla XML-muotoista, ja nykyään käytetäänkin enemmän JSON-muotoa, jota on helpompi käsitellä Javascriptissa.

Polyfill on termi, jolla tarkoitetaan standardin mukaisen natiivin selainominaisuuden toteuttamista Javascriptilla (tai mahdollisesti jollain muulla tekniikalla). Termillä tarkoitetaan yhteensopivuuskirjastoa, jolla saadaan toteutettua uusi standardin mukainen ominaisuus myös vanhemmissa selaimissa. Polyfill kirjasto saattaa esim. sovittaa selaimesta löytyvät selainkohtaiset ominaisuudet standardin mukaiselle rajapinnalle. HTML5:n kannalta merkittäviä Javascriptilla toteutettuja yhteensopivuus-

kirjastoja ovat esim. Modernizr ja HTML5shiv. Polyfill-termiä käytti ensimmäistä kertaa Remy Sharp kirjassaan: Introducing HTML5 [1]. Termiä ovat myös levittäneet Modernizr-kirjaston kehittäjät [2]. Modernizr on yhteensopivuuskirjasto, jonka tarkoituksena on tuottaa laajempi yhteensopivuus HTML5:lle ja CSS3:lle. HTML5shiv on keskittynyt enemmän takaamaan HTML5 yhteensopivuutta Internet Explorer selaimen versiolle 8 ja sitä vanhemmille.

2.3 CSS

CSS eli Cascading Style Sheets on dokumentti, jolla määritetään sivuston visuaaliseen ulkoasuun vaikuttavia ominaisuuksia, kuten esim. käyttöliittymäelementin värejä, kokoa tai tekstin fontteja sekä muita ominaisuuksia. CSS-määrityksiä voidaan kohdistaa tietyille elementtityypeille, tietyn luokan omaaville elementeille, yksittäiselle elementille ID:n perusteella sekä monilla muilla valitsimilla. CSS:ssä yksittäisen elementin yksittäiseen ominaisuuteen voi kohdistua useampia erilaisia määrityksiä, joista käytetään sitä, jolla on tarkin valitsin, esim. ID:een perustuva määritys ylikirjoittaa pelkän luokan perusteella määritetyn ominaisuuden. CSS ei ole varsinaisesti HTML:n osa vaan erillinen tekniikka, joka kehittyy omana osanaan. CSS3:a pidetään usein HTML5:een liittyvänä tekniikkana. Alla on yksinkertainen CSS-esimerkki, joka on tehty aikaisemmalle merkintäkielen esimerkille.

```
.paragtext {
    font-family: Verdana, Geneva, sans-serif;
}
#input1 {
    height: 2em;
}
select {
    width: 10em;
    height: 2em;
}
body {
    background-color: #e5f2f2;
}
```

Koodiesimerkki 3. Yksinkertainen CSS-dokumentti.

CSS dokumentti koostuu valitsimista, jonka jälkeen aaltosulkeiden sisällä määritetään ominaisuus ja ominaisuuden arvo. Esimerkiksi alussa oleva valitsin .paragtext kertoo, että asetus halutaan kohdistaa elementeille, joilla on luokka paragtext. Elementin luokat määritellään elementin class-attribuutissa, kuten aikaisemmassa HTML-

esimerkissä näytettiin. Seuraava valitsin (#input1) kertoo, että sääntö halutaan kohdistaa elementille, jonka id on input1. ID on HTML-merkinnässä attribuutti, jonka pitäisi olla yksilöllinen yhdelle elementille. kaksi viimeistä valitsinta kohdistaa säännöt tietyn tyyppisille elementeille (select ja body). CSS-määrityksiä voidaan kirjoittaa HTML-dokumentin sisälle, mutta ne on usein helpompi kirjoittaa kokonaan omaan dokumenttiinsa, johon HTML-dokumentti linkitetään.

2.4 Palvelinohjelmointi

2.4.1 PHP

PHP oli alkujaan vain kokoelma CGI-skriptejä, mutta on sittemmin laajentunut kokonaiseksi ohjelmointikieleksi, joka on 5-version alusta myös oliopohjainen. PHP:tä on mahdollista suorittaa mm. Linuxilla ja Windowsilla myös komentoriviltä, mutta yleisin käyttö PHP:lle on palvelinohjelmoinnissa. PHP:tä ei käännetä ennen palvelimelle asettamista kone- tai tavukoodiksi vaan palvelimen PHP-moduuli tulkitsee selväkielisen PHP-koodin, mikä tekee PHP-sovelluksen rakentamisesta helppoa, sillä koodia ei tarvitse jatkuvasti kääntää ja kopioida palvelimelle. PHP-kehitykseenkin löytyy useita kehitysympäristöjä, mutta sovelluksia on mahdollista luoda käyttäen tekstieditoria, verkkopalvelinta ja selainta. Jos PHP:ssä on asetettu päälle virheraportointi, niin syntaksivirheet ja poikkeukset näkyvät virheilmoituksina selaimessa, kun navigoidaan palvelimen sivustolle. Virheraportointi kannattaa aina asettaa pois päältä tuotantopalvelimelta, sillä muuten kuka tahansa sovelluksen käyttäjä voi virhetilanteessa saada virheilmoituksen, joka saattaisi paljastaa tietoturvan kannalta arkaluontoista tietoa. PHP-tiedostossa voidaan käyttää sekaisin staattista HTML:ää ja PHP-koodia. PHP toimii tavallisesti palvelinympäristössä niin, että selain pyytää palvelimelta PHP-tiedostoa. Palvelin ei lähetä varsinaista PHP-tiedostoa asiakkaalle vaan tulkitsee PHP-koodiosat ja lähettää lopuksi muodostuvan datan käyttäjälle. PHP on ottanut syntaksiinsa vaikutteita Javasta, C:stä ja Perlistä [3]. Alla on yksinkertainen esimerkki PHP:stä

```
<!DOCTYPE html>
<html>
    <head>
        <title>testi</title>
    </head>
<body>
    <?php
        echo (“helloworld”);
    ?>
</body>
</html>
```

Koodiesimerkki 4. Yksinkertainen PHP-esimerkki.

“<?php” ja “?>” merkkijonojen välissä olevat merkinnät tulkitaan PHP-koodiksi, joka käy palvelimella PHP-tulkin läpi, ja dokumenttiin tulostetaan koodin tilalle ainoastaan koodissa määritetyt tulosteet. Kuten esimerkistä huomaa niin dokumentti voi sisältää myös staattista HTML:ää. Sivuston ei tarvitse olla edes kovin suuri, jotta PHP:n ja HTML:n muodostava kokonaisuus menee sekavaksi. Tällaista tilannetta pyritään helpottamaan MVC-ohjelmistokehitysmallilla, jota tässä työssä käytetty Codeigniter-ohjelmistokehys toteutti.

2.4.2 Codeigniter

Codeigniter on avoimen lähdekoodin MVC-ohjelmistokehitysmallia toteuttava ohjelmistokehys (framework), jonka on kehittänyt EllisLab. Ellislabin ExpressionEngine-sisällönhallintajärjestelmä on myös rakennettu Codeigniterin päälle. Codeigniter löytyy myös github-sivustolta, jossa sovelluskehyksestä löytyy useita erillisiä kehitysversioita (fork). Ellislab ilmoitti vuoden 2013 syyskuussa etsivänsä uutta omistajaa Codeigniterille. Codeigniteria käyttäviä sivustoja ovat esim. AT&T Center (attcenter.com), gamestack.net ja sprintcenter.net. Sitepointin vuoden 2014 Best PHP-frameworks -kyselyssä Codeigniteri oli 4. sijalla Yii-frameworkin kanssa pitäen perää Laravel, Phalcon ja Symphony2 frameworkeille. Codeigniterissa on valittu lähtökohdaksi sovelluskehyksen keveys, mikä on saanut kehuja mm. PHP:n alkuperäiseltä kehittäjältä, Rasmus Lerdorfilta. Codeigniter pyrkii myös siihen, että sovellushehys olisi käytettävissä mahdollisimman monen hostingpalvelun kanssa, minkä takia PHP-tulkin lisäksi Codeigniter ei vaadi juuri mitään muuta, vaikka esim. Apachen kanssa onkin mahdollista käyttää mod_rewritea URL:n lyhentämiseen. Vaikka Codeigniteria on kehuttu sen lähtökohdista keveydelle ja yksinkertaisuudelle, on sitä myös kritisoitu siitä että se ei tarjoa monipuolisia rakenteita, kuten esim. Object Relational mappingia (ORM), jolla saadaan liitettyä tietokantataulun tiedot suoraan olioihin. Myös pitkää

tukea vanhemmille PHP-versioille on kritisoitu, sillä esim. PHP5-tuki ja sen mukanaan tuoma oliopohjaisuus on ollut Codeigniterissa vasta 2.x-versioissa.

Codeigniter toteuttaa MVC-sovellusmallia, joka on alunperin työpöytäsovelluksista lähtöisin oleva sovelluskehitysmalli, jonka tavoitteena on erottaa graafisen sovelluksen käyttöliittymä- ja logiikkaosat erilliseksi kokonaisuudeksi niin, että sovellus olisi helpommin hallittavissa, ja että sovelluksen käyttöliittymä tai logiikka voidaan vaihtaa niin ettei toista osaa tarvitse muuttaa. Codeigniter toteuttaa MVC-mallia suhteellisen joustavasti niin että esim. malleja ei ole pakko käyttää ollenkaan vaan sivun voi toteuttaa pelkästään ohjaimella ja näkymillä.

Codeigniterin tämän hetkinen versio 2.1.4 vaatii palvelimen, jonne on asennettu PHP 5.2.4 tai uudempi. Codeigniterin käyttöönotto verkkopalvelimelle, joka sisältää jo valmiiksi PHP-tulkin, ei vaadi muuta kuin codeigniterin lataamisen Ellislabin kotisivuilta ja tiedostojen kopioimisen palvelimelle. Mitään varsinaista asennusta ei tarvita, sillä Codeigniter koostuu pelkistä PHP-tiedostoista, joita palvelimen PHP-moduuli osaa tulkita. Codeigniter on kehittäjälle avointa koodia, johon on myös lisenssin nojalla mahdollisuus tuottaa omia muutoksia. Codeigniterissa kehittäjällä on käytössään kuusi erilaista komponenttia, jotka ovat MVC:n osat eli ohjain, malli, ja näkymä sekä “avustajat”(helper), kirjastot (library) ja konfiguraatiotiedostot. Kaikki Codeigniterin komponentit kirjoitetaan PHP:llä. Codeigniter kansiot sisältävät monien verkkopalvelinten käyttämiä .htaccess-konfiguraatiotiedostoja, jotka määrittävät Codeigniterin kansiot niin, ettei palvelin tarjoa niiden sisältöä verkkoon päin, mutta parhaan tietoturvan saadakseen kannattaa codeigniterin application- ja system-kansiot palvelimella jättää ulkopuolelle siitä kansiosta, jonka sisältöä tarjotaan verkkoon päin, jotta käyttäjillä ei ole mitään mahdollisuutta tehdä verkkopyyntöjä, jotka kohdistuisivat codeigniterin järjestelmäkansioihin ja niiden tiedostoihin.

Codeigniterissa on yksittäinen aloitustiedosto (index.php), joka suorittaa kaikki aloistustoimepiteet, minkä jälkeen suoritus etenee osoitteen määrittämään ohjaimeen tai konfiguraatioon asetettuun oletusohjaimeen. aloitustiedoston tulee olla ainoa PHP-tiedosto, jota käyttäjä voi palvelimelta pyytää suoraan. Kaikki muut sovelluskehykseen kytkettävien osien tulee olla sijoitettu palvelimelle niin ettei niihin ole suoraa pääsyä palvelimen ulkopuolelta. Erilliset CSS- ja Javascript-tiedostot on mahdollista sijoittaa palvelimelle niin että niihin on suora pääsy. Ohjaimen tehtävänä on ladata mallit, jotka suorittavat varsinaista sovelluslogiikkaa ja näkymät, jotka luovat käyttöliittymän, joka lähetetään asiakaspuolelle. Ohjelman suoritus alkaa käytännössä ohjaimesta, jossa

suoritetaan URL:n määrittämä funktio. Codeigniterin käyttämä URL on muodoltaan: Palvelimenosoite.fi/index.php/ohjain/funktio/lisäparametri. Apachella on mahdollista käyttää mod_rewritea niin että kaikki palvelimen pyynnöt ohjataan index.php:lle, jolloin URL:n ei tarvitse osoittaa kyseiseen tiedostoon. Funktiosta on mahdollista kirjoittaa useita versioita, jotka vastaanottavat eri määrän parametreja eikä funktion tarvitse vastaanottaa parametreja ollenkaan. Ohjaimeen on mahdollista kirjoittaa funktioita, joihin ei voi kohdistaa pyyntöjä. Tällaisen funktion nimi pitää alkaa kahdella alaviivalla (_).

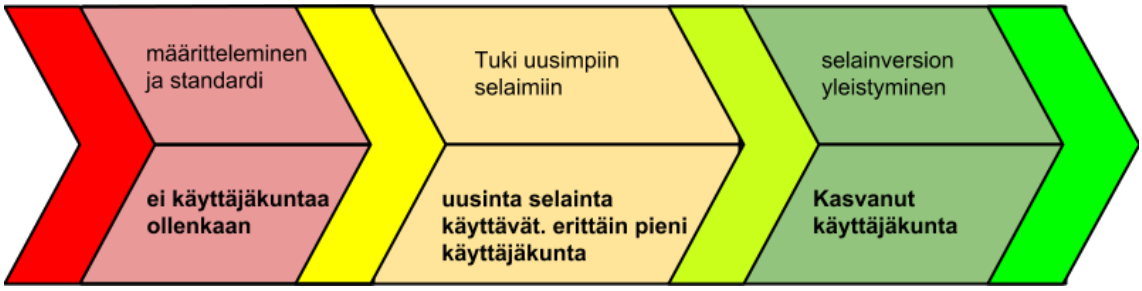
Codeigniterin näkymien on tarkoitus luoda käyttäjälle lähetettävä HTML. Näkymät ladataan ohjaimessa, ja ohjain voi ladata kokonaisen näkymän useassa osassa, jolloin myös näkymät ja HTML-runko voidaan pilkkoa pienempiin osiin. Näkymissä käytetään myös PHP-koodia, mutta tarkoituksena on PHP-koodissa ainoastaan tulostaa sivulle dynaamista dataa. Käyttöliittymäpuolella voi hyödyntää useita helpereitä, joiden tarkoitus on helpottaa esim. toistuvien listojen ja taulukoiden tulostamista sekä monia muita ominaisuuksia kuten CAPTCHA-nimellä tunnettu kuvahaasteeseen perustuva roskaviestien estojärjestelmä. Ohjain voi välittää näkymälle dataa, jota voidaan käsitellä näkymässä aivan kuten tavallisia muuttujia.

Codeigniter ei ole ainoastaan MVC-mallia toteuttava kehys vaan tuo kehittäjän käyttöön myös monia lisäominaisuuksia ja erillisiä sovellusrajapintoja ominaisuuksille, jotka ovat jo valmiina PHP:ssä. Codeigniterin käyttäjäistunnot eroavat PHP:n vakio istunnoista siinä, että Codeigniterin sessioita ei tallenneta verkkopalvelimelle vaan selainkeksiin salattuna tai salaamattomana ja vaihtoehtoisesti vielä jommankumman vaihtoehdon lisäksi relaatiotietokantaan, josta istunto varmennetaan. Käyttäjäistunnolle voidaan myös yksinkertaisesti konfiguroida käyttöön ip-osoiteen ja/tai user agentin varmistaminen, jolloin sessio hävitetään, jos seuraava pyyntö tulee eri IP-osoitteesta kuin mikä on alunperin merkitty käyttäjäistuntoon tai jos useragent tunniste ei vastaa aikaisempaa. Codeigniteria hyödyntävässä sovelluksessa on täysin mahdollista käyttää edelleen PHP:n toteutusta käyttäjäistunnoille. Jos verkkosovellus luodaan todella suuren käyttäjämäärän käytettäväksi, voi olla järkevää käyttää Codeigniterin sessiototeutusta, jonka ansiosta on mahdollista hajauttaa sovellus usealle sovellus-palvelimelle, koska käyttäjäistunnon tietoja ei tallenneta yksittäisen sovelluspalvelimen muistiin. Codeigniterin käyttäjäistunnoissa ei ole mahdollisuutta tallentaa varsinaisia sessiotietoja palvelimen muistiin, kuten PHP:n käyttäjä-istunnoissa.

2.5 Selaintekniikoiden kehittyminen ja käyttöönotto

Verkkotekniikat kehittyvät jatkuvasti ja tarve tuottaa verkkosivuille entistä enemmän monimuotoista sisältöä kasvaa. Standardit ovat oleellinen asia verkkotekniikoille, joiden pitäisi toimia usean sovellusvalmistajan asiakasohjelmilla ja useilla eri laitteilla. Verkkotekniikoita määrittelee World Wide Web Consortium eli lyhyemmin W3C. W3C tuottaa standardista määrittelydokumentteja, jotka etenevät useamman vaiheen läpi ennen kuin niistä tulee varsinaisia standardeja [4]. Working Draft on ensimmäinen julkisesti esiteltävä luonnos, jolle pyydetään julkista arviointia ja palautetta. Candidate Recommendation on vaihe, jossa määrittelyn tunnetaan olevan tarpeeksi valmis toteutettavaksi. Candidate Recommendation -asteella olevaan määritykseen varataan vielä mahdollisuus tehdä muutoksia perustuen toteutuksesta saatavaan palautteeseen, mutta mitään laajoja muutoksia ei ole enää tässä vaiheessa odotettavissa. Proposed Recommendation on yleensä vain muodollinen vaihe, jossa määrittelyn tekijät ja toteuttajat hyväksyvät määrittelyn eteenpäin vietäväksi, ja muutokset ovat entistä epätodennäköisempiä. W3C Recommendation tarkoittaa, että määritys on lopullinen ja siitä on tullut standardi. Vaikka standardit ovatkin oleellinen asia niin standardin määritykset ovat usein varsin huonoa opiskelumateriaalia verkkokehittäjälle, sillä määritykset ovat erittäin teknisiä ja laajoja sekä kohdistuvat verkkokehittäjän lisäksi myös selainvalmistajille ja mahdollisesti myös palvelinohjelmistojen valmistajille. Vaikka verkkokehittäjä pääosin hakeekin tietoa muista lähteistä kuin varsinaisen standardin määrittelydokumenteista, niin verkkokehittäjälle on kuitenkin hyödyllistä ymmärtää päällisin puolin prosessi, jonka viralliset määritykset käyvät läpi ja miten suuria muutostavoitteita liittyy kuhunkin vaiheeseen. Kun määritykset ovat saavuttaneet suhteellisen vakaan tilan, alkavat selainvalmistajat yleensä toteuttaa niitä. Selainvalmistajat tukevat toisinaan ominaisuuksia jo ennen kuin ne ovat saaneet standardin aseman.

Verkkosivustoja valmistavalle verkkokehittäjälle ei yleensä riitä se, että tekniikka on olemassa ja toteutettu, vaan tekniikalle pitää olla myös vastaanottajia. Seuraavan kuvan aikajana hahmottaa miten tekniikan käyttöönotto yleensä etenee.



Kuva 2. selaintekniikan aikajana

Mitään universaalia sääntöä tekniikan käyttöönoton aikataululle ei ole, mutta yleiseen käyttöön suunnatun verkkosivuston pitäisi toimia hyvin laajasti eri selainvalmistajien ohjelmilla tukien myös hieman vanhempia selainversioita.

Kun varsinainen standardi on valmistunut ja selaimet aloittaneet uusien ominaisuuksien tukemisen, kestää yleensä hetken ennen kuin käyttäjät ovat ottaneet käyttöön uusimmat selainversiot siinä määrin, että ominaisuuden käyttäminen on järkevää. Tärkeänä mittarina tällaisessa tilanteessa on selainten markkinaosuudet ja ehkä vieläkin tarkemmin selainversioiden osuudet. Selainten käyttöosuuksiin liittyviä tilastoja julkaisevat mm. Statcounter, Clicky, W3Counter sekä WebApplications. Näiden palveluiden tarjoamia lukuja kannattaa pitää vain suuntaa antavina arvioina, mutta ne antavat kuitenkin apua esim. tilanteessa, jossa harkitaan yhteensopivuuden tiputtamista tietylle vanhentuneelle selainohjelmalle.

Aivan jokaisen verkkotekniikan kohdalla ei kuitenkaan olla täysin riippuvaisia selainten käyttöönoton viiveestä. Jotkin ominaisuudet eivät välttämättä suoraan vaadi erillistä selaintukea ja joitain ominaisuuksia on mahdollista toteuttaa erilaisten Javascript-kirjastojen kautta.

3 HTML5-standardi

HTML5 on erittäin monimuotoinen käsite. HTML5 on nimi uudelle verkkotekniikka-standardille sekä merkintäkielelle, joka on tämän standardin osa. Standardi käsittää merkintäkielen lisäksi myös laajennuksia Javascript API:iin. HTML5-ominaisuuksista puhuttaessa tilanne on vieläkin sekavampi, koska HTML5:n kehityksen aikana useita HTML5-määritykseen kuuluneita ominaisuuksia on siirtynyt kokonaan omaksi osakseen muodostaen erillisen standardin, mutta näitä ominaisuuksia pidetään

yleensä silti HTML5:een liittyvinä tekniikoina kuten myös muutamia muita tekniikoita, jotka eivät varsinaisesti ole missään vaiheessa olleet osa HTML5:tä, mutta jotka ovat useissa yhteyksissä myös liitetty HTML5:een. HTML5:n kehitystä on myös pyritty nopeuttamaan ja joitain kaavailtuja ominaisuuksia on siirretty tulevaan HTML 5.1 -standardiin. Tässäkin työssä käydään läpi useita tekniikoita, jotka usein luetaan osaksi HTML5:ä, vaikka ne eivät virallisesti sitä olekaan. Käsittelen tämän osion aliotsikoissa useita HTML5-ominaisuuksia. Tämä teoreettinen osa ei kata kaikkia HTML5-tekniikoita, mutta käy läpi niistä tärkeimmät.

HTML-merkintäkielen aikaisemman standardiversion 4.01 jälkeen W3C lähti kehittämään XHTML2:sta, mikä ei kuitenkaan saanut kannatusta, sillä XHTML 2:ssa suunniteltiin taaksepäin yhteensopivuuden rikkomista ja XHTML 2 hylättiinkin myöhemmin ja pohjaksi päätettiin ottaa WHATWG-työryhmän esitys HTML5:stä. WHATWG eli Web Hypertext Application Technology Working Group on työryhmä, jonka perustivat useat henkilöt eri selainvalmistajilta. Työryhmä alkoi tuottamaan omaa HTML5-määritystään, jonka oli tarkoitus olla taaksepäinyhteensopiva vanhempien HTML-versioiden kanssa. W3C otti WHATWG:n tuottaman määrityksen vasta myöhemmin pohjaksi viralliselle standardille, mutta voidaan kuitenkin katsoa että HTML5:n kehittämisen alkaneen jo vuodesta 2004, jolloin WHATWG perustettiin. HTML5:lle on olemassa edelleen erikseen myös WHATWG:n määritys, joka sai myöhemmin nimekseen HTML Living Standard. W3C:n määritysdokumentti on näistä kahdesta virallisen standardin määritys.

HTML-merkintäkielen kehitys on ollut pitkä prosessi. Merkintäkielen aikaisempi versio ilmestyi jo vuoden 1999 joulukuussa. Monet verkon multimediaominaisuudet on jouduttu standardia odotellessa toteuttamaan standardiin kuulumattomilla tekniikoilla, kuten Adoben Flashilla, Microsoftin Silverlightilla tai muulla vastaavalla selainlaajennuksella. HTML5:n merkittävimpinä ominaisuuksina pidetäänkin mahdollisuuksia piirtää grafiikkaa tai näyttää videoita selaimessa ilman lisäosia. HTML5 ei kuitenkaan vielä mahdollista kaikkia ominaisuuksia, joita oli mahdollista hyödyntää esim. Flashin tai Silverlightin kautta, mikä on esim. videoiden kanssa aiheuttanut pientä kiistaa, eikä myöskään yhtenäisestä videoformaatista ole saatu sopua aikaiseksi. HTML, Javascript ja CSS ovat alkujaan olleet vain verkkosivun rakentamiseen tarkoitettuja työkaluja, mutta tekniikat ovat jo ennen HTML5:n tuloa alkaneet kehittyä siihen suuntaan että niillä on mahdollista toteuttaa paljon muutakin. HTML5:n myötä verkkotekniikoista toivotaan syntyvän kokonainen järjestelmäriippumaton ohjelmistoalusta.

HTML5 on keskeneräinen, mutta se on saavuttanut useita merkittäviä virstanpylväitä ja sen määrittämiä uusia tekniikoita hyödynnetään jo hyvin laajasti ympäri internetiä. Täydellistä takuuta sille, että kaikki määritykset pysyvät nykyisellään loppuun asti, ei ole, mutta on myös turha odottaa suuria muutoksia enää tässä vaiheessa, kun tavoitellaan, että määritys saataisiin vietyä standardiksi vuoden 2014 lopussa.

3.1 HTML5-Merkintäkieli

HTML5-merkintäkieli on yritetty pitää mahdollisimman samanlaisena kuin aikaisemmatkin versiot ja yhteensopivana vanhempienkin selainten kanssa. Aikaisemmat HTML-versiot perustuivat W3C:n SGML-merkintäkieleen, mutta HTML5 ei enää perustu tähän yleiseen merkintäkieleen, vaan määrittelee kokonaan oman syntaksinsa. Muutos näkyy esim. HTML5-dokumentin alussa, johon ei tarvitse enää kirjoittaa pitkänomaista DOCTYPE-määritystä. HTML5:n yksinkertainen dokumentti-määrityskin on olemassa HTML5:ssä oikeastaan vain sen takia, että sitä käyttävät sivustot toimisivat vanhemmissa selaimissa.

Pienistä muutoksista huolimatta varsinainen merkintäkieli ei ole aikaisemmista versioista muuttunut. HTML5-dokumentti koostuu edelleen elementtien muodostamasta puurakenteesta aivan kuten aikaisemmatkin HTML-versiot, ja kaikki aikaisemmat elementitkin ovat edelleen käytettävissä. HTML5-merkintäkielen uudistukset ovat uusissa elementeissä ja vanhojen elementtien laajennuksissa. Keskeisimpiä merkintäkielen ominaisuuksia ovat input-syötekenttien uudet tyyppimääritykset, video-elementti, audio-elementti, canvas-elementti sekä mahdollisus käyttää muutamia erillisiä merkintäkieliä, kuten MathML:ää, SVG:tä ja RDFa:ta HTML-merkintöjen sisällä. Merkintäkielen määritykset ovat kirjoitushetkellä W3C:n Candidate Recommendation asteella, mikä tarkoittaa että muutokset ovat vielä mahdollisia, mutta suuria muutoksia tuskin on odotettavissa. Candidate Recommendation tason dokumentti varoittaa muutaman ominaisuuden mahdollisesta poistumisesta, mikäli niitä ei aleta tukea. HTML5 puuttuu määrityksellisesti joidenkin vanhojen elementtien käyttöön, mutta standardi määrittää kuitenkin, että selainten pitää näyttää myös nämä vanhat elementit oikein.

3.1.1 Uudet rakenteelliset elementit

Verkkosivustoilla on yleensä tiettyjä rakenteita, kuten esim. navigointipalkki, ylätunniste, alatunniste ja sivun varsinainen sisältö. Tällaiset osiot luodaan yleensä käyttämällä div-elementtejä, joiden asemointi, koko ja tyyli määritetään CSS-dokumentissa. Tällaisella tavalla luodut sivun osiot toimivat tarkoitetulla tavalla, ja käyttäjä ymmärtää niiden merkityksen, mutta näin toteutetuilla sivuosioilla on tietyt haittapuolensa, joista merkittävin on se, että koneet eivät osaa erottaa sivuosien merkitystä. HTML5 määrittää useita rakenteelllisia elementtejä, jotka eivät toiminnaltaan eroa siitä, että käytettäisiin div-elementtejä, mutta uudet elementit antavat sivun osille tarkan merkityksen, mitä voidaaan hyödyntää esim. esteellisyystoiminnoissa. HTML5 tarjoaa mm. header- (ylätunniste), footer- (alatunniste), nav- (navigaatio), section- (osio) , aside- (sivuhuomautus) ja article-elementit (artikkeli), joita on tarkoitus käyttää sivuosien merkkaamiseen.

Uudet elementit eivät varsinaisesti luo mitään uudentyyppistä käyttöliittymää vaan niiden tarkoituksena on ainoastaan korvata vanha tapa käyttää yhtä ainoaa elementtiä sivuosien tuottamiseen. Vanhat selaimet eivät ymmärrä uusia rakenne-elementtejä, mutta ne osaavat kuitenkin näyttää niiden sisällön, jos se koostuu tunnetuista elementeistä. Vanhemmissa selaimissa ongelmaksi muodostuu tyylimääritysten kohdistaminen näille uusille elementeille, koska selain ei näitä elementtejä tunnista, niin ei selain myöskään osaa kohdistaa niille tyylimäärityksiä. Seuraavassa taulukossa näkyy selaintuki rakenteelllisille elementeille.

Taulukko 1. Selainyhteensopivuus rakenteellisille elementeille

Selain	**Lähtien versiosta**	**version julkaisuaika(kk/vvvv)**
Apple Safari	5	6/2010
Internet Explorer	9	3/2011
Mozilla Firefox	4	3/2011
Google Chrome	6	9/2010
Android Browser	2.2	5/2010
Internet Explorer Mobile	10	10/2012
Opera	11.1	12/2010

Jos haluaa käyttää uusia rakenteellisia elementtejä ja taata sivustolle laajempaa yhteensopivuutta, voidaan Javascriptissa muuttaa uudet rakenteelliset elementit div-elementeiksi silloin, kun käyttäjällä on käytössään selain, joka ei tue uusia elementtejä. Yhteensopivuutta varten voidaan käyttää myös jotain valmista yhteensopivuuskirjastoa esim. html5shiv.

3.1.2 MathML

MathML on XML:ään perustuva merkintäkieli, jolla voi tuottaa matemaattisia yhtälöitä ja muita matemaattisia merkintöjä verkkosivuille. MathML on erillinen määrittely, joka luetaan HTML5:een liittyväksi tekniikaksi. MathML on saavuttanut virallisen suosituksen aseman. HTML5:ssä MathML-merkintää voi tuottaa HTML-dokumenttiin math-elementin sisälle. Selain tuki MathML-merkintäkielelle on erittäin vajaata.

Taulukko 2. selainyhteensopivuus MathML-merkinnälle

Selain	**Lähtien versiosta**	**version julkaisuaika(kk/vvvv)**
Apple Safari	5.1	7/2011
Internet Explorer	ei tukea	---
Mozilla Firefox	2.0	10/2006
Google Chrome	ei tukea*	---
Android Browser	ei tukea	---
Internet Explorer Mobile	ei tukea	---
Opera	ei tukea	---

Chrome tuki ominaisuutta hetkellisesti versiossa 24 mutta otti sen myöhemmin pois päältä epävakauden takia. MathML:ää voi hyödyntää natiivia selaintukea laajemmin käyttämällä esim. MathJax-kirjastoa.

3.1.3 Uudet lomakekentät

Input-käyttöliittymäelementti on saanut uusia tyyppejä. Aikaisemmin kyseinen elementti on toiminut lähes pelkästään tekstikenttänä. HTML5:ssä kentän saa määriteltyä esim. numero- tai emailtyyppiseksi. kannattaa huomata, että kirjoitushetkellä HTML5-standardi on vielä kesken ja W3C:n Candidate Recommendation -asteen

dokumentissa varoitetaan esimerkiksi, että color-tyyppi saatetaan jättää pois vähäisen tuen takia, vaikka se tällä hetkellä määrittelyistä löytyykin. Mikäli olemassa oleva sivusto käyttää jotain Javascript-toteutusta eri tyyppisille input-kentille, niin sivuston ei kannata ainakaan vielä siirtyä käyttämään pelkästään uusia input-tyyppejä. Selaintuki uusille lomakekentille on melko vähäistä, mutta uusia tyyppejä voi kuitenkin käyttää huoletta, sillä vanhemmat selaimet antavat tavallisen tekstikentän, jos tyypin määrittävää type-attribuuttia ei tunnisteta [5, s. 144].

Input-kentälle löytyy myös uusi attribuutti nimeltä placeholder, jonka arvo näkyy tekstikentässä, kunnes käyttäjä kirjoittaa jotain kyseiseen lomakekenttään. Aikaisemmin tieto siitä, mitä tietoa kenttään halutaan, on yleensä kerrottu vain tekstillä tai label-elementillä. Placeholderia voi käyttää suhteellisen turvallisesti, sillä sivusto näkyy muuten oikein vaikka selain ei osaisikaan näyttää placeholderin määrittämää tekstiä. Käyttöliittymä kannattaa kuitenkin rakentaa niin, että placeholderit eivät ole ainoa opaste lomakekenttien täyttämiselle, jotta vanhempien selainten käyttäjätkin saavat ohjeet lomakekenttien täyttämiseen [5, s. 156].

3.1.4 Data-attribuutit

Data-attribuutit ovat HTML5:n ominaisuus, mutta ne toimivat periaatteessa missä tahansa selaimessa, joka tukee DOM:n ensimmäistä tasoa, joka on esim. Internet Explorerissakin tuettu jo versiosta 5 lähtien. Data-attribuutit ovat käytännössä vain nimiavaruussääntö, jolla varataan kehittäjille tila tehdä omia tietokenttiään HTML-merkintöjen sisään. Data-attribuutteja on tarkoitus käyttää ainoastaan Javascriptista eikä niiden ole tarkoitus määrittää mitään metadataa (kts. Metadata). Ominaisuus on hyödyllinen esim. tietokannan surrogaattiavaimien sisällyttämiselle. Data-attribuutit ovat aivan tavallisia attribuutteja, mutta niiden pitää aina käytää attribuutin nimessä alkuosaa ‘data-’. Jos dokumentti käyttää HTML5:n dokumenttityyppiä, niin HTML5-validaattori hyväksyy data-attribuutit validina merkintänä. HTML5 lisää javascriptiin dataset-ominaisuuden, jolla on helpompi hakea ja käydä läpi data-attribuutteja.

Taulukko 3. Selainyhteensopivuus datasetille

Selain	**Lähtien versiosta**	**version julkaisuaika(kk/vvvv)**
Apple Safari	5.1	07/2011
Internet Explorer	11	10/2013
Mozilla Firefox	6.0	08/2011
Google Chrome	7.0	10/2010
Android Browser	3.0	02/2011
Internet Explorer Mobile	Ei tukea	---
Opera	11.01	01/2011

Selaintuki Datasetille vaihtelee, mutta data-attribuutit on täysin mahdollista noutaa ja asettaa Javascriptissa vanhoilla DOM-metodeilla getAttribute ja setAttribute, joita tukevat käytännössä kaikki selaimet.

3.1.5 Drag and Drop

Drag and Drop nimellä kutsuttu ominaisuus mahdollistaa käyttöliittymä elementtien raahaamisen ja tiputtamisen toisten elementtien päälle. Elementeille on mahdollista määrittää draggable-attribuutti ja Javascriptin puolelle on määritetty useita tapahtumia elementtien raahaamiseen ja tiputtamiseen. Drag and Drop on osa virallista HTML5-määritystä ja kirjoitushetkellä Candidate recommendation -asteella. Drag and Drop perustuu Microsoftin API:iin. Seuraavassa taulukossa näkyy selaintuki Drag and Dropille.

Taulukko 4. Selainyhteensopivuus drag and dropille.

Selain	**Lähtien versiosta**	**Version julkaisuaika(kk/vvvv)**
Apple Safari	3.1	03/2008
Internet Explorer	5.5*	07/2000
Mozilla Firefox	3.5	06/2009
Google Chrome	4.0	01/2010
Android Browser	Ei tukea	---
Internet Explorer Mobile	10	10/2012
Opera	12.0	06/2012

Drag and Drop -ominaisuus perustuu Microsoftin määritykseen, joka on ollut olemassa Internet Explorerissa jo kauan aikaa, mutta standardin Drag and Dropissa on joitain lisäkohtia, joita IE ei tue vielä kirjoitushetkelläkään (IE11). Esimerkiksi dataTransfer-olion getData- ja setData-metodilla on Internet Explorerissa mahdollista käyttää vain text- tai URL-avainta vaikka HTML5-määrityksessä on mahdollista käyttää useita muitakin tyyppejä [7].

3.1.6 Vektorigrafiikka

SVG eli Scalable Vector Graphics on W3C:n määritys, joka ei ole varsinaisesti HTML5-standardin osa, mutta joka yleensä luetaan HTML5:een liittyväksi tekniikaksi. HTML5 on tuonut merkintäkieleen svg-elementin, jonka avulla HTML-merkintöjen sisään voi tuottaa SVG-merkintää. Kannattaa huomioida, että vaikka SVG-standardi on ollut valmiina jo vuodesta 2001 lähtien, niin HTML5:n svg-elementti kuuluu HTML5-standardiin, joka on kirjoitushetkellä Candidate Recommendation -asteella. SVG-standardin ensimmäinen versio julkaistiin jo vuonna 2001, mutta selaimet ovat alkaneet vasta myöhemmin tukea laajasti SVG-formaattia. SVG on XML:ään perustuva tiedostomuoto, jota voidaan käyttää verkkosivulla HTML5:n svg-elementin lisäksi tavallisissa kuvissa tai sivun taustakuvana. Varsinaisia tiedostoja voidaan käyttää joko tekstimuodossa (svg-tiedostopääte) tai tekstimuotoinen tiedosto voidaan pakata gzip-pakkausalgoritmilla (svgz-tiedostopääte). Vektorigrafiikan ero tavallisiin kuviin nähden on se, että vektorigrafiikka ei koostu kuvapisteistä (pikseleistä) vaan koordinaatistoon piirretyistä muodoista, joita voidaan skaalata todella suurta näyttötarkkuutta (resoluutio) käyttäviin laitteisiin tai laitteisiin, joilla on erittäin pieni näyttötarkkuus. SVG-tuki on

moninainen asia, sillä selain voi tukea HTML5:n svg-elementtiä tai tiedostojen käyttämistä sivun taustakuvissa tai img-elementeissä.

Taulukko 5. Selainyhteensopivuus HTML5:n svg-elementille

Selain	**Lähtien versiosta**	**version julkaisuaika(kk/vvvv)**
Apple Safari	5.1	07/2011
Internet Explorer	9	03/2011
Mozilla Firefox	4.0	03/2011
Google Chrome	7.0	10/2010
Android Browser	3.0	02/2011
Internet Explorer Mobile	10	10/2012
Opera	11.60	12/2011

3.1.7 Video- ja audioelementti

HTML5 mahdollistaa äänen ja videon sisällyttämisen verkkosivustolle audio- ja videoelementillä. Standardi ei määritä videolle eikä audiolle mitään tiettyä formaattia, jota kaikkien selainten pitäisi tukea, minkä takia media pitää yleensä liittää useammassa eri formaatissa, jotta se olisi kaikkien käyttäjien saatavissa. Video voidaan näyttää lähes kaikilla selaimilla, jos video tarjotaan sekä H.264-formaatissa, että jossakin rojaltivapaassa formaatissa (Ogg ja WebM). Apple ja Microsoft tukevat selaimissaan lähes yksinomaan H.264:ää, kun taas kaikki avoimen lähdekoodin selaimet (esim. firefox ja konqueror tukevat lähes aina Ogg- tai WebM-formaattia. Audioformaattien puolella tilanne on hieman parempi sillä lähes jokainen selain tukee mp3-formaattia. Osa selaimista (esim. Firefox) tukevat MP3-formaattia järjestelmään asennettujen erillisten koodekkien kautta. Monet selaimet osaavat käyttää videon ja audion toistamiseksi erillisiä koodekkeja niin että selain voi toistaa muitakin videoformaatteja kuin ne, joita selain tukee natiivisti. Video- ja audio-elementissä on mahdollista määrittää useampi lähde kyseiselle multimedialla, jolloin multimedia voidaan liittää useassa eri formaatissa ja näin saada video tai audio toimimaan suurimmalla osalla selaimista. Alla on esimerkki video-elementin käytöstä.

```
<video height="480" width="720" controls>
    <source src="video.ogg" type=”video/ogg”/>
    <source src=”video.mp4” type=”video/mp4” />
    Selaimesi ei tue HTML5-videota
</video>
```

Kuvan merkintä sisältää huomautustekstin, joka näkyy käyttäjille, joiden selain ei tue HTML5-videota. Audio-elementillä on samankaltainen muoto, jota en tässä käy sen enempää läpi. Javascriptissa on mahdollista tarkistaa tukeeko selain käytettävää formaattia. Seuraavassa taulukossa näkyy selaintuki video-elementille.

Taulukko 6. Selainyhteensopivuus HTML5:n video-elementille.

Selain	**Lähtien versiosta**	**version julkaisuaika(kk/vvvv)**
Apple Safari	4.0	6/2009
Internet Explorer	9	3/2011
Mozilla Firefox	3.5	6/2009
Google Chrome	4.0	1/2010
Android Browser	2.3	12/2010
Internet Explorer Mobile	10	10/2012
Opera	10.5	3/2010

Taulukossa mukana olevien selainten osalta tuki audio-elementille on sama. HTML5-videot eivät tue kirjoitusvaiheessa läheskään kaikkia niitä ominaisuuksia, joita on ollut mahdollista käyttää Flashilla ja Silverlightilla, mutta jos sivustolla ei tarvita perusominaisuuksien lisäksi mitään erityistä niin HTML5-videoilla on mahdollista korvata erillisillä laajennuksilla tuotettava videotoisto. HTML5-videoon on ehdotettu useita laajennuksia, joilla voidaan toteuttaa monia niistä ominaisuuksista, joita hyödynnetään tällä hetkellä selainlisäosilla. Media Source extensions -määrityksen (MSE) tarkoituksena on mahdollistaa esim. videolähteen vaihtaminen yhteysnopeuden vaihdellessa. MSE:tä ovat ehdottaneet Google, Microsoft ja Netflix. Encrypted Media Extensionsin (EME) tarkoituksena on määritellä miten selain kommunikoi erillisen sisällönpurkumoduulin kanssa ja näin mahdollistaa esim. videoiden suojaamisen DRM-teknologialla. Avoimen lähdekoodin tukijat ovat vastustaneet raskaasti DRM:n sisällyttämistä standardiin. W3C päätti kuitenkin syyskuussa 2013 ettei EME ole aihealueen ulkopuolella ja onkin mahdollista että EME tulee olemaan standardin osa tulevaisuudessa.

3.1.8 Canvas

Canvas on sivulla näkyvä piirtoalue, jolle voidaan Javascriptin avulla tuottaa grafiikkaa dynaamisesti. Canvasille voidaan tuottaa myös 3D-grafiikkaa käyttämällä WebGL:ää. Canvas on osa HTML5-määritystä, joka on kirjoitushetkellä Candidate Recommendation asteella. Canvasille on kehitteillä useita laajennuksia, kuten fullscreen API ja Text API, jotka ovat kirjoitushetkellä vasta Working Draft -asteella. Seuraavassa taulukossa näkyy selaintuki Canvas-elementille ja samaan määritykseen kuuluville Javascript toiminnoille.

Taulukko 7. Selainyhteensopivuus HTML5:n canvasille.

Selain	**Lähtien versiosta**	**version julkaisuaika(kk/vvvv)**
Apple Safari	3.1	03/2008
Internet Explorer	9	03/2011
Mozilla Firefox	2.0	10/2006
Google Chrome	4.0	01/2010
Android Browser	3.0*	02/2011
Internet Explorer Mobile	10	10/2012
Opera	9.0	03/2009

Android tuki suurimmilta osin canvasia jo 2.1 versiossa, mutta ei toDataurl-metodia

3.1.9 Metadata

Metadata on ”tietoa tiedosta”. Metadata kuvaa ihmiselle esitettävästä datasta käsitteitä ja asiasuhteita niin, että ne ovat tulkittavissa myös ohjelmallisesti. Esimerkki metadatasta voisi olla esim. tilanne, jossa tekstin pätkässä mainitaan useita henkilöitä ja metadatalla merkitään tekstistä pätkät, jotka viittaavat henkilöihin.

HTML5:ssä on kaksi eri metadata-formaattia: RDFa (Resource Description Framework in attributes) ja Microdata. Kumpikin formaatti käyttää attribuuttilaajennuksia asioiden kuvaamiseen sekä erillisiä sanastoja, joissa määritetään erilaisia käsitteitä. Jos metadataa tuotetaan sivustolle hakukoneita varten, on hyvä käyttää jotain yleisessä käytössä olevaa sanastoa. RDFa:ta tukevat hakutuloksien jäsentelyssään ainakin Google ja Yahoo. Metadata ei varsinaisesti vaadi erillistä selaintukea ja olemassa

olevien merkintäkielten sääntöjen mukaan kirjoitettavat attribuuttilaajennukset eivät aiheuta selaimissa poikkeustilanteita eikä selaimen tarvitse ymmärtää metadatan sisältöä.

3.1.10 Vanhentuneet merkinnät

Joitain vanhoja elementtejä ja attribuutteja on HTML5:ssa merkitty “vanhentuneeksi” (obsolete). Kaikki tällaiset merkinnät toimivat edelleen selaimissa ja standardi määrittää myös, että niitä tulee tukea, mutta samaan aikaan halutaan myös että verkkokehittäjät lopettaisivat näiden ominaisuuksien käyttämisen. Osa vanhentuneista merkinnöistä saa aikaan validaattorissa vain varoituksen, jossa opastetaan verkkokehittäjää käyttämään vaihtoehtoista tapaa ko. Ominaisuudelle ja osa vanhentuneista merkinnöistä antaa validaattorissa suoraan virheen, kuten esimerkiksi Javasovelluksen liittämiseen käytettävä applet-elementti, joka on korvattu embed- tai object-elementillä, joilla on tarkoitus hoitaa kaikkien erillisten teknologioiden (esim. Flash, Silverlight ja Java) liittäminen verkkosivustolle. HTML5:n mukana ei kuitenkaan ole varsinaisesti poistunut (deprecated) mikään vanha merkintätapa[8]. En käy tässä tarkemmin läpi vanhentuneita merkintöjä. Lista kaikista vanhentuneista HTML-merkinnöistä löytyy W3C:n määrittelyistä osoitteesta: http://www.w3.org/TR/html5/obsolete.html

3.2 HTML5 Javascript-API

HTML5 laajentaa tuntuvasti Javascript-API:a. Uusiin ominaisuuksiin lukeutuvat mm. Server-sent events ja Web storage. Monille uusille Javascript ominaisuuksille ei kuitenkaan vielä löydy tukea kaikista selaimista. Kehittäjän on kuitenkin mahdollista tarkistaa Javascript-koodissa, tukeeko selain ominaisuutta. Useimmissa tapauksissa uudet Javascript-ominaisuudet on tehty omiin luokkiinsa ja yhteensopivuuden selaimen kanssa voi testata tarkistamalla tunnistaako selain tarvittavaa luokkaa. Tässä osiossa ei ole tarkoitus näyttää kattavaa listaa uusista Javascript-ominaisuuksista vaan käsitellä päällisin puolin merkittävimmät uudet ominaisuudet.

3.2.1 WebSocket

WebSocket on tekniikka, jolla voidaan muodostaa avoin kaksisuuntainen yhteys, jonka läpi palvelin voi lähettää käyttäjälle dataa tai toisinpäin. WebSocket käyttää TCP-

protokollaa. Mikäli selain ei tue WebSocketia ei juuri ole mahdollista saada toimintaan edes millään polyfill-kirjastolla (vrt. Server-sent events). WebSocket on kirjoitushetkellä Candidate Recommendation asteella. WebSocket sisälsi aikaisemmassa määrittelyssään tietoturva-aukon, joka määrityksisssä paikattiin. Jotkin selaimet toteuttivat haavoittuivaista määritystä, mutta tämä on myöhemmissä versioissa päivitetty. Seuraavassa taulukossa näkyy selaintuki WebSocketille.

Taulukko 8. Selainyhteensopivuus WebSocketille

Selain	**Lähtien versiosta**	**version julkaisuaika(kk/vvvv)**
Apple Safari	6	7/2012
Internet Explorer	10	10/2012
Mozilla Firefox	11*	3/2012
Google Chrome	14.0	9/2011
Android Browser	4.4	10/2013
Internet Explorer Mobile	10	10/2012
Opera	12.10	11/2012

Firefox tuki WebSocketia jo versiossa 4 ja 5 mutta ominaisuus oli oletuksena pois päältä koska siihen liittyi sinä aikana tietoturvaongelmia. Firefoxin versioilla 6-10 ominaisuus on käytettävissä, mutta se on käyttänyt näissä versioissa Firefoxin omaa mozwebsocket-laajennusta. Standardin määrittämä luokan nimi on ollut Firefoxissa käytössä versiosta 11 lähtien. Opera tuki myös tietoturvaongelmia sisältävää määritystä versiossa 11, mutta se oli Firefoxin tavoin myös oletuksena pois päältä.

3.2.2 Server-sent events

Server-sent events (tästä eteenpäin SSE) mahdollistaa myös erillisen yhteyden luomisen palvelimeen. WebSocketiin verrattuna SSE ei ole aivan yhtä monipuolinen. Yhteysprotokollana käytetään HTTP:tä, ja SSE:ssä varsinaista dataa lähetetään vain käyttäjän suuntaan. SSE on tarkoitettu erityisesti sisällönpäivitykseen ja soveltuu siihen paremmin kuin WebSocket. SSE:tä ei tueta kaikilla selaimilla, mutta monia vanhempia selaimia varten voi käyttää jotain polyfill-kirjastoa, jolla SSE saadaan toimimaan niiden kanssa. Server Sent Events on W3C:n Candidate Recommendation -asteella, joten muutokset ovat vielä mahdollisia. Tällä hetkelläkin on olemassa useita toteutuksia, joilla saadaan pidettyä palvelimen ja selaimen välinen HTTP-yhteys avonaisena, mutta

vasta SSE:n tai Websocketin myötä tällaiselle toiminnallisuudelle on standardinmukainen toteutustapa. Seuraavassa taulukossa näkyy selain tuki SSE:lle

Taulukko 9. Selainyhteensopivuus Server Sent Eventsille

Selain	**Lähtien versiosta**	**version julkaisuaika(kk/vvvv)**
Apple Safari	5.0	06/2010
Internet Explorer	ei tukea	----
Mozilla Firefox	6	08/2011
Google Chrome	6.0	09/2010
Android Browser	4.4	10/2013
Internet Explorer Mobile	ei tukea	----
Opera	11.0	12/2010

Ominaisuutta ei siis tällä hetkellä tueta Internet Explorerin millään versiolla luultavasti johtuen siitä, että SSE ei vielä ole saavuttanut varsinaisen standardin asemaa. Ominaisuutta pystyy kuitenkin hyödyntämään jopa Internet Explorerilla käyttämällä jotain polyfill-toteutusta.

3.2.3 Geolocation

Geolocation ei virallisesti ole HTML5:n osa vaan kokonaan erillinen määrittelynsä, mutta se luetaan usein HTML5:een liittyväksi tekniikaksi. Geolocation on saavuttanut jo standardin aseman. Geolocation määrittelee tavan käsitellä verkkosovelluksessa paikannustietoja ja sen ansiosta verkkosovellus voi pyytää käyttäjän selaimelta käyttäjän tarkkaa sijaintia ja verkkosivusto voi toteuttaa palveluita, jotka hyödyntävät sijaintitietoja. Selain pyytää aina käyttäjältä luvan sijainnin käyttämiseen ennen kuin Javascript-sovellus saa käyttöönsä paikannustietoja. Seuraavassa taulukossa näkyy selaintuki Geolocationille.

Taulukko 10. Selainyhteensopivuus geolocationille

Selain	**Lähtien versiosta**	**version julkaisuaika(kk/vvvv)**
Apple Safari	5	06/2010
Internet Explorer	9.0	03/2011
Mozilla Firefox	3.5*	06/2009
Google Chrome	5.0	05/2010
Android Browser	2.1	01/2010
Internet Explorer Mobile	10	10/2012
Opera	10.6	07/2010

Firefox on tukenut geolocationia jo versiossa 3.5, mutta ennen versiota 24 geolocationin olemassaoloa testaava koodi (kts. Seuraava koodiesimerkki) palautti tosi-arvon, vaikka geolocation olisi otettu pois päältä. Seuraavassa koodiesimerkissä näkyy, miten Geolocationia käytetään.

```
if("geolocation" in navigator) {
    // selain tukee geolocationia
    var geo_asetukset = {
        enableHighAccuracy : true,
        maximumAge : 0
        timeout : 15000
    };
    navigator.geolocation.getCurrentPosition(successfunction
            ,errorfunction
            ,geo_asetukset);
}
else {
    alert("Selaimellasi ei voi käyttää paikannustietoja");
}
```

getCurrentPosition-funktio toimii asynkronisesti eli funktiolle annetaan viittaus funktioon, jota se kutsuu kun sijainnin selvittäminen on tehty. Funktiolle voi lisäksi antaa lisäparametreina virhetilanteessa käytettävän funktion ja asetusolion, jossa voidaan määritellä esim. tarkan sijainnin selvittäminen käyttöön. Selain pyytää käyttäjän luvan sijaintitietojen käyttöön ennen kuin sijainti luovutetaan Javascript-sovellukselle.

3.2.4 Selector API

Selector API on laajennus DOM:iin. Selector API mahdollistaa elementtien hakemisen samanlaisilla valitsimilla kuin CSS:ssä. Tällaisia ominaisuuksia on aikaisemmin käytetty

pääasiassa jollain Javascript-kirjastolla, kuten JQuery. Selector API:n 1. taso on saanut virallisen suosituksen aseman ja sillä on suhteellisen hyvä selaintuki. Selector API:n 2. taso on siirretty osaksi DOM level 4 -määritystä. Seuraava koodiesimerkki näyttää, miten Selector-API:a voidaan käyttää.

```
// tyyppi nodeList
var elementit = querySelectorAll(”.luokka”);
```

Koodiesimerkki 5 . Yksinkertainen esimerkki selector-API:n käytöstä.

Koodiesimerkissä noudetaan CSS:n luokkavalitsimella kaikki ne elementit, joiden class-attribuutin arvo on luokka. Seuraavassa taulukossa näkyy selaintuki Selector API:lle

Taulukko 11. Selainyhteensopivuus Selector API:lle

Selain	**Lähtien versiosta**	**version julkaisuaika(kk/vvvv)**
Apple Safari	5.1	07/2011
Internet Explorer	8*	03/2009
Mozilla Firefox	3.5	06/2009
Google Chrome	4.0	01/2010
Android Browser	2.1	01/2010
Internet Explorer Mobile	10	10/2012
Opera	10.0	09/2009

Jokaisella selaimella on käytettävissä vain ne CSS-valitsimet, joita selain muutenkin tukisi, mistä syystä IE8 tukee vain CSS2.1 -valitsimia. IE9 tukee myös CSS3-valitsimia.

3.2.5 WebGL

WebGL ei ole osa W3C:n HTML5-standardia, eikä W3C edes hallinnoi WebGL:ää, mutta se luetaan usein HTML5 liittyväksi tekniikaksi. WebGL on OpenGL ES 2.0:aan perustuva grafiikka-API Javascriptille. WebGL:ää kuten myös OpenGL:ää ylläpitää Khronos Group. WebGL:n avulla on mahdollista piirtää 3D-grafiikkaa canvas-elementin sisälle hyödyntäen asiakaskoneesta löytyvää 3D-grafiikkapiiriä. Seuraavassa taulukossa näkyy selaintuki WebGL:lle.

Taulukko 12. Selainyhteensopivuus WebGL:lle.

Selain	**Lähtien versiosta**	**version julkaisuaika(kk/vvvv)**
Apple Safari	5.1	07/2011
Internet Explorer	11	10/2013
Mozilla Firefox	4.0	03/2011
Google Chrome	8.0	12/2010
Android Browser	ei tukea*	---
Internet Explorer Mobile	ei tukea	---
Opera	12.0	06/2012

Androidilla jotkin puhelinvalmistajat ovat tukeneet erillisillä firmware-päivityksillä WebGL:ää[9], mutta Android Browserilla ei ole virallista tukea WebGL:lle. Chromen Android-versio tukee WebGL:ää.

3.2.6 Web Workers

Web Workers -nimellä tunnettu ominaisuus mahdollistaa skriptien suorittamisen eri säikeissä niin, että Javascript-sovellukset pystyvät paremmin hyödyntämään moni-ydinprosessoreita. Ominaisuus on kirjoitushetkellä W3C:n Candidate Recommendation -asteella, joten ominaisuuden määrityksellä on vielä mahdollisuus muuttua.

Taulukko 13. Selainyhteensopivuus Web Workersille

Selain	**Lähtien versiosta**	**version julkaisuaika(kk/vvvv)**
Apple Safari	4.0	02/2009
Internet Explorer	10	10/2012
Mozilla Firefox	3.5	06/2009
Google Chrome	4.0	01/2010
Android Browser	4.4*	10/2013
Internet Explore Mobile	10	10/2012
Opera	10.6	07/2010

Android Browserissa oli hetkellisesti käytössä Web Workers jo versiossa 2.1, mutta se poistettiin 2.2-versiossa, koska Androidin käyttämä Javascript-moottori ei tukenut

säikeiden vaatimia lukituksia. Web workersin sisällä suoritettava sovelluskoodi ei pääse suoraan muokkaamaan verkkosivun sisältöä, vaan se voi ainoastaan lähettää viestejä skriptille, joka toimii käyttöliittymäpuolella.

3.2.7 Web Storage

Verkkosovelluksella ei ole ollut kovin suuria mahdollisuuksia aikaisemmin tallentaa dataa asiakaspuolelle. Evästeet ovat olleet olemassa jo pitkään, mutta niiden käyttö soveltuu vain tiettyihin tilanteisiin, ja evästeen koko on erittäin rajattu (vain 4 kilotavua). Web Storage mahdollistaa huomattavasti suuremman datamäärän tallentamisen asiakaspuolelle. Evästeen data lähetetään aina automaattisesti HTTP-pyynnössä palvelimelle, kun taas Web Storageen asetettu tieto voidaan Javascriptissa lähettää vain tarpeen mukaan. Web Storageen tallennetaan avain-arvopareja. Tieto tallennetaan Web storagessa merkkijonomuodossa, mikä tarkoittaa sitä että suuremmat datarakenteeet pitää serialisoida esim. JSON:ksi. Web Storagessa on kaksi eri tyyppistä tietovarastoa. Localstorage tallentaa tiedon pysyvästi, mikä tarkoittaa sitä, että tieto säilyy, kunnes esim. käyttäjä tyhjentää manuaalisesti selaimeen tallennetun datan. Selain pitää huolta siitä, että Web Storageen sijoitettu data on vain tallennuksen tehneen domainin saatavilla. Localstoragea ei kannata käyttää silloin, jos saman verkkonimen alla toimii useampia erillisiä sivustoja. Sessionstorage tallentaa datan vain sen hetkisen selainistunnon ajaksi, ja data häviää, kun selain suljetaan. Seuraavassa taulukossa näkyy selaintuki Web Storagelle.

Taulukko 14. Selainyhteensopivuus Web Storagelle

Selain	**Lähtien versiosta**	**version julkaisuaika(kk/vvvv)**
Apple Safari	4.0	06/2009
Internet Explorer*	9	03/2011
Mozilla Firefox	3.5	6/2009
Google Chrome*	5	05/2010
Android Browser	2.0	10/2009
Internet Explorer Mobile	10	10/2012
Opera	10.50	3/2010

Internet Explorer tuki Web storagea jo osittain versiossa 8, mutta vasta 9-versiossa löytyi tuki storage-eventeille. Chrome tuki jo 4-versiossa localStoragea mutta vasta 5-

versiossa sessionStoragea. Seuraava koodinpätkä sisältää esimerkin Web Storagen käyttämisestä.

```
if(typeof(Storage) !== "undefined") {
    localStorage.muuttuja = "lokaaliarvo";
    sessionStorage.muuttuja = "sessioarvo";
    alert(localStorage.muuttuja + " " + sessionStorage.muuttuja);

}
else {
    alert("Selaimesi ei tue HTML5 Web storagea");
}
```

Koodiesimerkki 6. Perusesimerkki Web Storagen käytöstä

Esimerkissä tarkistetaan alussa typeof-operaattorilla että alusta tuntee Storage-luokan. koodinpätkä tallettaa tietoja sekä pidempiaikaiseen tallennustilaan (localStorage) että istuntomuistiin (sessionStorage).

4 Verkkosovelluksen tietoturva

Verkkosovelluksiin liittyy monia yleisiä tietoturvauhkia, jotka esiintyvät jatkuvasti tietomurtojen yhteydessä. Käytännön työn kannalta tietoturva oli erittäin oleellinen asia, sillä sovelluksen kautta olisi mahdollista hallita täysin palvelun käyttäjiä ja tarkastella sovelluksen lokitiedostoja. Tässä osiossa käsitellään yleisellä tasolla tietoturvaa liittyen verkkosovelluksiin ja erityisesti PHP:n kannalta. Open Web Application Security Project (OWASP) julkaisee listaa yleisimmistä sivustojen tietoturvaongelmista. Listan kärkipäässä on lähes poikkeuksetta aina Injection(kataa esim. SQL injectionin), XSS ja puutteellinen käyttäjäsession hallinta. Nämä haavoittuvuudet olivat myös OWASP:n vuoden 2013 listan kärjessä[6]

4.1.1 PHP:n tietoturva

PHP liitetään usein huonoon tietoturvaan, mutta varsinaiseen PHP-tulkkiin suoraan liittyvät tietoturvaongelmat eivät ole erityisen yleisiä. PHP sisälsi ennen versiota 4.2 ominaisuuden nimeltä register_globals, joka mahdollisti verkkopyynnössä saatujen parametrien lisäämisen suoraan koodissa oleviin muuttujiin. Jos register_globals oli päällä, ja kehittäjä ei alustanut tärkeitä muuttujiaan, pystyi hyökkääjä asettamaan muuttujan arvon verkkopyynnössä. Versiossa 4.2 ominaisuus asetettiin vakiona pois

päältä, versiossa 5.3 ominaisuutta on alettu poistaa käytöstä (deprecated) ja versiossa 5.4 se on poistettu kokonaan. Ensimmäisissä tietokantafunktioissa ei ollut mahdollisuuksia kirjoittaa valmisteltuja kyselyitä vaan käyttäjää vaadittiin erillisellä funktiolla käsittelemään parametrit SQL-injektion varalta, ja jos tätä ei muistettu tehdä jokaiselle tietokantahaun parametrille niin sovellukseen jäi SQL-injektion mahdollistava tietoturva-aukko. PHP on erittäin helposti omaksuttava kieli, mutta siinä ei aina kädestä pitäen varmisteta, että tietyt perustietoturvaan liittyvät asiat hoidetaan asianmukaisesti.

4.1.2 XSS-hyökkäys

XSS-hyökkäyksessä (Cross Site Scripting) käyttäjä lisää sivustolle lähetettävään dataan HTML- ja Javascript-merkintää. Mikäli palvelin lisää tämän datan suoraan sivulle näkyviin suodattamatta HTML:n erikoismerkkejä, niin käyttäjien selaimet suorittavat tämän koodin luottaen siihen, ettei palvelin lähetä itsellensä vahingollista dataa. Varsinainen hyökkäys vaatii yleensä että sivusto näyttää yksittäisen käyttäjän hyökkäysviestin muille käyttäjille, mutta vaikka tällainen tapaus ei olisikaan kyseessä, kannattaa käyttäjän lähettämä data kuitenkin käsitellä, ettei käyttäjän viesteissä ole erikoismerkkejä, jotka sotkevat HTML-merkintää ja saattavat jopa sekoittaa hänelle itsellensä näkyvää käyttöliittymää. Kaikille HTML-kielen erikoismerkeille on vaihtoehtoinen merkintätapa, jolla erikoismerkki on mahdollista näyttää tekstinä eikä HTML-merkintänä. Yksinkertaisimmassa XSS-viestissä on vain HTML:n script-elementti ja Javascript-koodia. Haitallinen viesti voidaan PHP:ssä torjua esim. ajamalla merkkijono htmlspecialchars- tai htmlentities-funktion läpi, jolloin erikoismerkit (esim. ‘<’ ja ‘>’) korvataan niiden vaihtoehtoisella esitystavalla (esim. ‘<’ ja ‘>'), jolloin viesti pysyy samana, mutta selain näyttää erikoismerkit tekstinä, eikä yritä tulkita niitä osana HTML:ää.

4.1.3 SQL-injektio

SQL-injektio liittyy myös käyttäjäsyötteiden tarkistusten laiminlyömiseen. SQL-tietokannnasta haetaan ja muokataan tietoja käyttämällä SQL-kyselykieltä. Jos kyselymerkkijonoon liitetään suoraan käyttäjän lähettämä parametri, niin käyttäjän on mahdollista lisätä kyselyn sekaan omia tietokantakomentojaan, joiden avulla käyttäjä saattaisi ohittaa kirjautumisen tai saada käsiinsä koko järjestelmän tietokannan. SQL-injektiota voidaan torjua käyttämällä esirakennettuja kyselyitä (Prepared Statement), joissa tietokannalle lähetetään kyselyn runko ja parametrit erikseen. Muita tapoja, joilla

SQL-injektiota voidaan torjua, ovat Stored Procedures ja perinteinen tapa, jossa parametrit käsitellään erillisellä funktiolla, joka korvaa erityismerkit vaihtoehtoisella esitystavalla ja suodatetut syötteet lisätään suoraan osaksi tietokantakyselyä. SQL-injektion ja muiden tietokanta-haavoittuvuuksien varalta on hyvä luoda tietokannalle useita käyttäjiä, joilla on kullakin pienimmät tarvittavat oikeudet tiettyyn tietokannan osaan.

PHP:ssä on useita sovellusrajapintoja tietokantojen käsittelyyn. Mysqli ja PDO mahdollistavat esivalmisteltujen kyselyiden kirjoittamisen. Vanhassa mysql-API:ssa on mahdollista suorittaa vain kokonaisia kyselyitä ja parametrit pitää käsitellä erillisellä funktiolla. Mysql-API on poistumassa PHP:stä.

4.1.4 Salasanojen suojaaminen

Salasanoja ei koskaan tule tallentaa selväkielisenä vaan käyttää tiivisteitä. Tiivistealgoritmi muodostaa sille annetusta syötteestä tietyn mittaisen sekoitetun merkkijonon, josta ei tavanomaisesti voida selvittää alkuperäistä syötettä. Käyttäjän salasana tarkistetaan kirjautumisen yhteydessä luomalla annetusta salasanasta tiiviste ja vertaamalla sitä tietokantaan tallennettuun tiivisteeseen. Mikäli tietokanta ikinä päätyy tietomurtajille, ei näillä ole suoraan käytössään jokaisen käyttäjän tunnus-salasana yhdistelmää, joilla mahdollisesti päästään käsiksi käyttäjän tileihin muilla sivustoilla, jotka käyttävät samaa tunnus-salasana yhdistelmää. Tiivisteitä käytettäessä myöskään ylläpito ei voi selvittää käyttäjän salasanaa, joten jos käyttäjä unohtaa salasanansa pitää salasana vaihtaa kokonaan uuteen.

Kuten aikaisemmin todettu tiivisteestä ei ole mahdollista suoraan selvittää alkuperäistä syötettä, mutta käyttämällä ns. rainbowtable-hyökkäystä on mahdollista selvittää tiettyjen sanojen tuottama tiiviste, jolloin yksinkertaiset salasanat on mahdollista selvittää. Rainbowtable on pitkä taulukko, johon on koottu syötteistä muodostuvia tiivisteitä. Tällaista salasanan selvittämistä voidaan estää suolaamalla tiiviste. Suola on pitkä satunnainen merkkijono, joka tallennetaan tietokantaan muiden käyttäjätietojen kanssa. Tiiviste-algoritmin syötteeksi luodaan merkkijono, johon on yhdistetty suola ja salasana. Suola pidentää syötettä, joka estää kokonaisen käyttäjätietokannan avaamisen rainbowtable -hyökkäyksellä. Nykyaikaisella tietokonelaitteistolla voidaan erittäin nopeasti luoda taulukko kaikista syöte-tiivistepareista, jotka on luotu lyhyestä

syötteestä. Käyttäjiä ei kuitenkaan voi pyytää lisäämään salasanojensa pituutta tekniikan kehittymisen tahdissa, joten syötettä pidennetään suolaamalla.

PHP:ssä on hash-funktio, jolla on mahdollista käyttää useita eri tiivistealgoritmejä, kuten esim. MD5, whirlpool, sha1, sha512 jne. Tiivistealgoritmin valintaa kannattaa myös harkita koska tilanne voi tietokoneiden laskentatehon ja haavoittuvuuksien takia muuttua. MD5 ei ole enää turvallinen käytettäväksi sertifikaattien kanssa. MD5:sta ei ole löydetty haavoittuvuutta, joka vaarantaisi salasanatiivisteet, mutta voi olla silti parempi käyttää jotain tuoreempaa algoritmia.

4.1.5 Session kaappaus

Kun käyttäjä kirjautuu verkkosivustolle, avataan käyttäjäistunto (sessio), jossa pidetään muistissa tieto, että käyttäjä on kirjautunut sivulle, eikä hänen tarvitse antaa kirjautumistietojaan uudelleen kuin vasta siinä vaiheessa, kun käyttäjäistunto raukeaa. Käyttäjäistunnon toteutuksia on useita, mutta jokaisessa toteutuksessa käyttäjän pitää lähettää jokin tunnistetieto palvelimelle. Hyökkäystapa, jossa pyritään selvittämään käyttäjäistunnon tunniste, jotta voidaan tekeytyä toiseksi käyttäjäksi, kutsutaan session kaappaamiseksi (Session hijacking). Oleellinen riski session kaappaamiselle on, jos käyttäjä on yhteydessä internetiin esim. julkisen salaamattoman Wlan yhteyden välityksellä, eikä kohdesivusto käytä salausta, jolloin kaikki käyttäjän lähettämä data on salaamatonta ja helposti salakuunneltavissa.

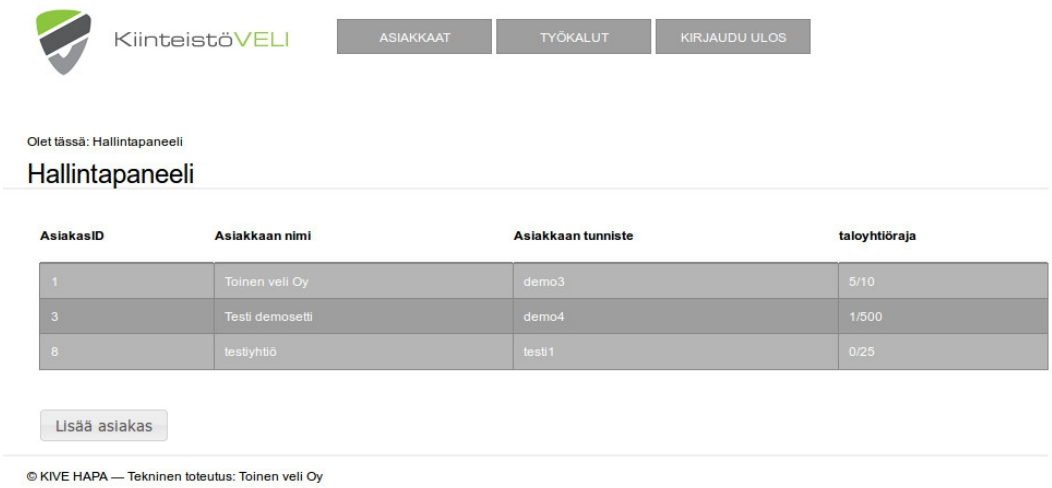
PHP:ssä on oma toteutuksensa käyttäjäsessioille, joka tallentaa selaimen evästeeseen sessiotunnisteen ja tallentaa palvelimelle varsinaisen istuntoon liittyvän datan. PHP-Session tietoturvaa parantaa oleellisesti sessiotunnisteen vaihtaminen jokaisen pyynnön yhteydessä, mitä ei PHP:ssä tehdä automaattisesti. Käyttäjäistunnon varmentamiseen voidaan käyttää sessiotunnisteen lisäksi myös muita tietoja. Sivupyynnössä näkyy selaimen User agent -tunniste, joka voidaan liittää istunnon tietoihin kuten myös IP-osoite, josta kirjautumisen tehnyt pyyntö on lähtöisin. Jos käyttäjän tiedot seuraavassa pyynnössä eivät vastaa istunnon aloitusvaiheessa muistiin otettuja tietoja, niin jätetään käyttäjän pyyntö käsittelemättä ja tuhotaan käyttäjäistunto. IP-osoitteen käyttäminen session varmennuksessa voi vaikeuttaa sovelluksen käyttöä niiden käyttäjien osalta, joilla IP-osoite vaihtuu usein. User agent -tunnisteen käytöllä ei ole varsinaisia negatiivisia vaikutuksia, mutta salaamatonta

yhteyttä käytettäessä myös käyttäjän useragent-tunniste on nähtävissä salaamattomassa verkkopyynnössä.

4.1.6 Cross Site Request Forgery

Cross site Request Forgery (tästä eteenpäin CSRF) on hyökkäysmuoto, jossa hyödynnetään yleensä toisen sivuston heikkoa tietoturvaa, jotta saadaan käyttäjän selain lähettämään kohdesivustolle verkkopyyntö, joka muokkaa tietoja kohdejärjestelmässä. Jos käyttäjä on kirjautunut kohdesivustolle niin käyttäjän selain lähettää pyynnön mukana myös sivustolle kuuluvat selainkeksit, joiden mukana kulkee yleensä myös käyttäjäistunnon tunniste. Yksinkertaisimmissa tapauksissa hyökkäys voi olla toteutettu niin että ulkopuoliselle sivustolle lisätään kuva, jonka lähteeksi merkitään osoite kohdesivustolla. Käyttäjän selain sivua ladatessaan tekee automaattisesti pyynnön osoitteeseen, joka on määritetty kuvan lähteeksi. Tällaisessa hyökkäyksessä voidaan suorittaa vain GET-tyyppinen HTTP-pyyntö, joka voi välittää dataa vain osoitteen rungossa. CSRF-hyökkäystä voi torjua useilla tavoilla. Kun Selain pyytää sivun osana olevaa kuvaa tai avaa linkin, niin pyynnön mukana lähetetään referrer-tieto, jossa selain kertoo mikä sivusto viittasi tekemään tämän pyynnön. Palvelin voi tarkistaa jokaisen dataa muokkaavan pyynnön yhteydessä, ettei pyynnön mukana ole selaimen lähettämää referrer-tietoa, joka osoittaisi jonkin muun sivuston ohjanneen selaimen tekemään ko. pyynnön. Vielä turvallisempi tapa torjua CSRF-hyökkäystä on tallentaa käyttäjäistuntoon sattumanvaraisesti luotu varmistuskoodi, joka lisätään jokaiselle sivulle, jolla käyttäjä voi muokata dataa ja tämä varmistuskoodi lähetetään kaiken muun datan mukana ja jos se ei vastaa käyttäjäistunnossa olevaa, niin ei muutoksia toteuteta.

5 Tuotettu työ



Kuva 3. Sovelluksen etusivu kirjautumisen jälkeen

5.1 Codeigniterin hyödyt

Codeigniter toi suuria hyötyjä tehdylle työlle verrattuna tilanteeseen, jossa sovellus olisi rakennettu täysin alusta asti itse. Codeigniter tarjosi järkevän rakenteen, jolle oli olemassa selkeä dokumentaatio. Tietoturvan puolelta tunnen, että codeigniter paransi tilannetta mahdollistamalla tarkistusten keskittämisen niin, että toistuvat tarkistukset, jotka pitää suorittaa joka kerta, tehdään automaattisesti jokaisen pyynnön yhteydessä, jolloin jokaisen sivun osan ei tarvitse erikseen suorittaa näitä tarkistuksia. Sovelluskehys konfiguroitiin käyttämään myös globaalia XSS-suodatusta, jolloin syötteistä ei tarvinnut erikseen suodattaa HTML:n erikoismerkkejä (XSS-hyökkäys). Tuotettuun kokonaisuuteen on mahdollista lisätä uusia osia (näkymiä ja malleja) niin, että nämäkin osat ovat edellä mainitun suojan takana. Codeigniteriin pääsi sisälle helposti, mutta tunnen silti, että vaikein puoli sen käyttöönotossa oli ymmärtää sen rakenne, ja miten ohjelman suoritus etenee kehyksessä. Dokumentaatio oli selkeää ja tiivistä, mutta se ei aina vastannut kaikkiin esille nousseisiin kysymyksiin.

Sovelluskehyksen keskeinen idea on mahdollistaa selkeä rakenne myös sen takia, että koodi olisi kehitystiimin osalta helpommin hallittavissa, mutta tällaisen tapauksen arviointi on tämän työn osalta vaikeaa, koska tuotin sovellusta yksin. Uskon kuitenkin, että Codeigniterin tarjoama dokumentoitu MVC-rakenne hyödyttäisi myös kokonaista ohjelmistokehitystiimiä. Codeigniter tarjosi myös oman sovellusrajapintansa monille sellaisille ominaisuuksille, joille löytyy valmis toteutuksensa myös valmiiksi PHP:stä.

Codeigniterin Database-luokkaa käytettiin tietokannan käsittelemiseen. Codeigniterin Database-luokka ei tarjonnut varsinaisesti mitään erityisiä lisäominaisuuksia PHP:n Data Objects -API:iin (PDO) verrattuna, mutta dokumentaatio oli mielestäni huomattavasti selkeämpi kuin PHP:n kotisivujen dokumentaatio PDO:sta. Joissain ominaisuuksissa pitäydyttiin kuitenkin PHP:n natiivitoteutuksissa. Esim. Codeigniterin käyttäjäistuntoja ei voinut käyttää samalla tavalla kuin PHP:n tarjoamia käyttäjä-istuntoja, jotka todettiin paremmin sopivaksi käytössä olleelle järjestelmä-kokoonpanolle.

5.2 HTML5 hyödyt ja käyttömahdollisuudet

Kaikissa käyttäjille lähetetyissä HTML-dokumenteissa käytettiin HTML5-dokumentti-tyyppiä. Varsinaisia HTML5-ominaisuuksia ei lopulta hyödynnetty erittäin laajasti, mutta suurin syy tähän oli enemmänkin tehdyn työn konteksti, joka ei hyötynyt esimerkiksi multimedia- ja grafiikkaominaisuuksista eikä hakukoneille suunnatuista metatiedoista. Työssä kuitenkin hyödynnettiin uusia input-elementin tyyppejä ja placeholdereita sekä data-attribuutteja. Kaikilla käytetyillä ominaisuuksilla on selvä taaksepäin yhteensopivuus vanhempien selainten kanssa. Uusista input-kentän tyypeistä käytettiin vain number- ja email-tyyppiä. Mahdollisuus esim. Selector-API:n käyttämiselle olisi ollut, mutta koska työssä käytettiin myös jQuery-javascriptkirjastoa, joka tarjosi Selector-API:a vastaavat ominaisuudet, ei Selector-API:n käyttäminen ollut varsinaisesti järkevää tässä tapauksessa. Työssä ei käytetty HTML5:n rakenteellisia elementtejä, koska sovelluksen oli tarkoitus toimia Internet Explorer 8 -tasoisella selaimella ja uusien elementtien käyttäminen olisi vaatinut Javascriptissa suoritettavia erillisiä yhteensopivuustoimenpiteitä, jotta käyttöliittymä olisi näkynyt oikein myös vanhemmilla selaimilla. Muutamien ominaisuuksien käytöstä piti siis jättäytyä, mutta muutoin ei paljolti törmätty tilanteisiin, joissa uutta ominaisuutta ei olisi ollut mahdollista käyttää. Ulkoisesti HTML5:n hyödyntäminen ei tässä työssä tuottanut muuta kuin joitakin pieniä käyttöliittymäparannuksia selaimille, jotka ko. ominaisuuksia tukevat ja teknisesti lähes samanlainen kokonaisuus olisi ollut mahdollista tämän työn tapauksessa toteuttaa vanhemmallakin HTML-versiolla, mutta työn aikana tuli kuitenkin selväksi, että HTML5:n käyttö oli täysin mahdollista haittaamatta sovelluksen taaksepäinyhteensopivuutta.

5.3 Javascript ja HTML5-yhteensopivuuskirjastot

Työssä hyödynnettiin Javascriptia suhteellisen laajasti. Käytössä oli JQuery-kirjasto, joka on tullut tunnetuksi siitä, että se mahdollistaa HTML-elementtien valitsemisen käyttäen CSS-valitsimia, mitä on alettu kehittämään myös verkkostandardeihin (Selector API). Elementtien valitseminen CSS-valitsimilla olisi onnistunut myös standardin mukaisella Selector API:lla, mutta päätin, että käytän JQueryn tarjoamaa toteutusta ominaisuudelle, koska se tarjosi laajemman taaksepäinyhteensopivuuden, ja oli itselleni tutumpi toteutus. Pienissä määrin käytettiin myös JQuery UI -kirjastoa, jolla tuotettiin useita käyttöliittymän osia, lähinnä erillisiä dialogeja, jotka avautuivat napin painalluksella. Tietojen lähetys hoidettiin pääasiassa AJAXilla käyttäen JQueryn tarjoamaan API:a. Rakennetun käyttöliittymän vahvuutena oli se, että selain ei vaihtanut sivua tiedon lähetyksen jälkeen, jos tiedon lähetys epäonnistui tai tiedoissa oli jokin virhe, jonka takia palvelin ei voinut hyväksyä tietoa. Virheen sattuessa käyttäjälle näytettiin virheilmoitus samalla sivulla, jolle tiedot syötettiin, ja kaikki käyttäjän syöttämät tiedot olivat edelleen tallella. Vaikka joitakin käyttöliittymäosia toteuttiin Jquery UI -kirjastolla pyrittiin kuitenkin siihen ettei Javascriptkoodi paisu liian suureksi. Työssä ei toteutettu esim. animaatioita käyttöliittymävalikoihin. Tarkoituksena oli Javascriptin käyttämisestä huolimatta pitää sivusto suhteellisen kevyenä, mikä ei olisi toteutunut, jos sivu olisi käyttänyt todella paljon Javascriptia.

Päädyin myös ratkaisuun olla käyttämättä HTML5:lle tarkoitettuja yhteensopivuus-kirjastoja. HTML5:n rakenteelliset elementit, joita päätin olla käyttämättä yhteen-sopivuusongelmien takia, olisi saatu toimimaan erittäin helposti käyttämällä Modernizr- tai HTML5shiv-kirjastoa, mutta en nähnyt tässä tapauksessa syytä käyttää näitä kirjastoja sillä kyseessä oli ainoastaan yksi pienimuotoinen ominaisuus, jolla ei olisi tämän työn kannalta suurta merkitystä ja lisäkirjastot vain lisäisivät Javascriptkoodin määrää. Jos työssä olisi hyödynnetty useampia HTML5-ominaisuuksia, olisin ottanut käyttöön jonkin yhteensopivuuskirjaston, joka olisi ratkaissut yhteensopivuuteen liittyviä ongelmia.

5.4 CSS:n käyttö ja toteuttaminen

CSS on ollut tämän työn käsittelyssä pienimmälle huomiolle jäänyt osuus. Työlle ei määritelty tiukkoja säädöksiä liittyen ulkoasuun, mutta toiveena oli, että se olisi tyyliltään samanlainen kuin KiinteistöVELI ja käytössä olivat KiinteistöVELI-palvelun

käyttämät CSS-määritykset. Valmiit CSS-määritykset toimivat apuna, mutta tälle työlle kuitenkin toteutettiin alusta asti oma käyttöliittymänsä, jolle määritettiin mm. asemointiin liittyvät ominaisuudet tilannekohtaisesti. Selainyhteenspivuuteen liittyvät vaatimukset vaikuttivat myös CSS:n käyttöön, minkä takia työssä ei juurikaan hyödynnetty CSS3-tason ominaisuuksia, joista osaa eivät tue edes tämän hetken uusimmat selaimet. Vaikka verkkoselain osaakin ohittaa tuntemattomat CSS-ominaisuudet, ei silti olisi ollut järkevää hyödyntää ominaisuuksia, jotka on tarkoitettu kohdeselainta uudemmille versioille. Mobiilikäyttöliittymä oli määritysten ja aihealueen ulkopuolella, enkä lähtenyt sitä toteuttamaan, mutta Media Queries -ominaisuuden avulla olisi ollut mahdolllista toteuttaa työhön erilliset tyyli- ja asemointisäädökset mobiililaitteille niin, että sovelluksen käyttö olisi helpompaa myös laitteilla, joissa on pieni näyttö.

Lähteet

1 Sharp, Remy. What is a polyfill. 2010 <http://remysharp.com/2010/10/08/what-is-a-polyfill/>. Luettu 14.4.2014.

2 Polyfills and Modernizr. Verkkodokumentti <http://modernizr.com/docs/#polyfills>. Luettu 14.4.2014.

3 PHP: preface. Verkkodokumentti <http://fi2.php.net/manual/en/preface.php>. Luettu 14.4.2014.

4 W3C Technical Report Development Process. 2005. Verkkodokumentti <http://www.w3.org/2005/10/Process-20051014/tr#maturity-levels>. Luettu 14.4.2014.

5 Korpela, Jukka K. 2011. HTML5 Uudet ominaisuudet. Jyväskylä: WSOYpro.

6 OWASP TOP 10 2013. 2013. Verkkodokumentti. <http://owasptop10.googlecode.com/files/OWASP%20Top%2010%20-%202013.pdf>. 14.4.2014.

7 Sharp, Remy. Native Drag and Drop. 2009. <http://html5doctor.com/native-drag-and-drop/>. Luettu 14.4.2014.

8 Louis Lazaris. About Obsolete features in HTML5. 2011. <http://www.impressivewebs.com/obsolete-features-html5/>. Luettu 14.4.2014.

9 Nilsson, Tobias. Xperia phones first to support WebGL. 2011. <http://developer.sonymobile.com/2011/11/29/xperia-phones-first-to-support-webgl/>. Luettu 14.4.2014.