Haaga-Helia
ammattikorkeakoulu Oy

Skaalautuva vektorigrafiikka mobiiliselaimissa

Mihkel Lind

Opinnäytetyö
Tietojenkäsittelyn koulutusohjelma
2016

Tiivistelmä

Haaga-Helia ammattikorkeakoulu Oy

Tekijä(t)
Lind Mihkel

Koulutusohjelma
Tietojenkäsittely

Opinnäytetyön otsikko
Skaalautuva vektorigrafiikka mobiiliselaimissa

Sivu- ja liitesivumäärä
24 + 12

Tässä opinnäytetyössä tutkittiin miten skaalautuvaa vektorigrafiikkaa (SVG) implementoidaan verkkosivulle ja minkälainen selaintuki sillä on mobiililaitteissa. Tutkimuksen ulkopuolelle rajattiin javascripti -kieli sekä SVG:n animointi.

Teoriaosuudessa kuvattiin mitkä ovat SVG:n etuja bittigrafiikkaan nähden sekä mainittiin SVG:n eri versioiden erot. Lisäksi käsiteltiin SVG:n ominaisuudet ja rakenne sekä sen keskeiset elementit ja attribuutit. Teoriaosuudessa käytiin myös läpi kuinka SVG koordinaatisto toimii ja mitkä tekijät siihen vaikuttavat. SVG:n implementointi esiteltiin yleisimpien tapojen osalta ja lisäksi esitettiin tapoja toteuttaa responsiivinen SVG -kuva.

Empiriaosuudessa kuvattiin mitä, millä, miten ja missä selaintukea testattiin. Testaus käsitti kuusi erilaista SVG:n käyttötapaa, joita testattiin kolmella fyysisellä mobiililaitteella. Lopulliset tulokset saatiin havainnoimalla kuvakaappauksia ja vertaamalla niitä teoriaan.

SVG:n selaintuen todettiin olevan hyvällä tasolla perusominaisuuksien osalta myös mobiiliselaimissa. Perimmäinen johtopäätös oli, että SVG on laiteriippumaton kuvamuoto, sillä se tuottaa suorituskykyistä ja tarkkaa kahdensuuntaista grafiikkaa kaiken tyyppisille laitteille. Tutkimuksen tuloksena syntyi teoriaan ja empiriaan pohjautuva tiivis ohjeisto SVG:n käytöstä verkkosivulla.

Asiasanat
SVG, selaimet, mobiililaitteet, XML, verkkosivut

Sisällys

Lyhenteet ja käsitteet

Adobe Illustrator
Vektorigrafiikan piirtämiseen ja muokkaamiseen käytettävä graafinen ohjelmisto. Sitä pidetään yhtenä edistyneimmistä ja yleisimmistä työkaluista vektorigrafiikan käsittelyssä, niin ammattilaisten kuin harrastajienkin keskuudessa.

Bittigrafiikka
Digitaalinen kuvamuoto, joka koostuu pikseleistä, ja jonka yleisimpiä tiedostomuotoja ovat PNG, JPG/JPEG ja GIF. Koska bittigrafiikka koostuu pikseleistä, sitä ei voi suurentaa loputtomiin ilman että kuvan laatu kärsisi.

SVG -attribuutti
SVG -elementin ominaisuus, jonka avulla annetaan lisäohjeita elementille. Attribuutin avulla voidaan määrittää esimerkiksi elementin koko, väri ja sijainti.

SVG -dokumentti
SVG tiedosto, jota käytetään tekstieditorin avulla. SVG -dokumentti on sama tiedosto kuin SVG -kuva. Se millä ohjelmalla sitä käsitellään, määrää sen esittämismuodon. Teksti- ja koodieditorit näyttävät SVG tiedoston XML muodossa.

SVG -kuva
SVG -tiedosto, jota käytetään graafisen käyttöliittymän avulla. SVG -kuva on sama tiedosto kuin SVG -dokumentti. Se, millä ohjelmalla sitä käsitellään, määrää sen esittämismuodon. Selaimet sekä vektorigrafiikka ohjelmistot näyttävät SVG tiedoston kuvana.

CSS
Muotoilukieli, jota käytetään HTML -dokumentin visuaalisen kerroksen luomiseen. CSS:n avulla määritetään HTML -elementtien ominaisuus kuten väri, koko tai sijainti. CSS -tyylimäärittelyjä voidaan käyttää myös SVG:n esittämiskerroksen attribuuttien määrittämiseen.

HTML
Verkkosivun sisällön kuvaamiseen ja esittämiseen käytettävä kieli. Sen viimeisimmän version, HTML5, avulla voidaan verkkosivulla monimuotoisempaa grafiikan käyttöä.

Laite
Tässä työssä laitteella tarkoitetaan yleisesti tietokonetta. Se voi työasema (pöytäköne), taulutietokone tai älypuhelin.

Mobiiliselain

Älypuhelimen tai tabletin ohjelmisto, jonka avulla selataan internetsivuja. Yleisesti käytösssä olevia mobiiliselaimia ovat Chrome (Android -laitteilla), Microsoft Edge Mobile, Internet Explorer Mobile sekä Mobile Safari.

Moderni Web -kehitys

Tässä työssä moderni Web -kehitys tarkoittaa laiteriippumattoman verkkosivun kehitystä. Kaikenkokoisille laitteille mukautuva verkkosivu on keskeinen osa modernia Web -kehitystä.

SVG

Skaalautuva vektorigrafiikka. Kuvaformaatti, joka on samalla XML-dokumentti. Sen avulla voidaan tuottaa verkkosivulle terävintä mahdollista grafiikkaa. Sitä voidaan suurentaa loputtomiin ilman että kuva menettää terävyyttään.

W3C

World Wide Web Consortiumin. Kanainvälinen yhteisö, joka kehittää Web -teknologioita. Sen jäseniä ovat mm. Opera Software, Mozilla Foundation, Microsoft Corporation, jotka ovat näin ollen mukana kehittämässä myös SVG:tä.

XML

Kuvauskieli, jonka avulla säilötään ja siirretään tietoa. Se on suunniteltu luettavaksi niin ihmisille kuin tietokoneillekin. SVG hyödyntää XML-kuvauskieltä.

1 Johdanto

Yksi modernin Web -kehityksen tavoitteista on laiteriippumattoman verkkosivun suunnittelu ja toteutus. Laitteiden näytön tarkkuudesta johtuen bittigrafiikkakuvat, kuten PNG ja JPG/JPEG, saattavat menettää terävyytensä. Skaalautuvan vektorigrafiikan (SVG) ansiosta voidaan verkkosivulle toteuttaa grafiikkaa ja kuvaa, jotka pysyvät terävänä päätelaitteen näytön tarkkuudesta riippumatta. (Soueidan 2015). SVG on ollut yleisimmissä internetselaimissa tuettuna jo useamman vuoden, mutta vasta nyt se käyttö alkaa yleistyä. SVG -kuvan implementointia verkkosivulla täytyy kuitenkin tutkia ja selaintukea testata, sillä selaintuki ei ole täysin aukoton. Vaikka jokin selain tukeekin SVG:tä laajasti, tietyn SVG:n ominaisuudet voivat käyttäytyä odottamattomalla tavalla (Coyer 2016, 4).

Tässä työssä tutkittiin SVG:n implementointia verkkosivulla ja mobiiliselainten tarjoamaa tukea SVG:lle. Teoriaosuudessa käsiteltiin SVG:n koordinaatistoa ja keskityttiin sen keskeisimpiin elementteihin ja esittämiskerroksen attribuutteihin. Keskeiset elementit valittiin sen mukaan, miten usein niitä esitettiin lähdemateriaalin esimerkeissä. Keskeiset attribuutit olivat taas keskeisten elementtien useimmiten käyttämiä attribuutteja lähdemateriaalin esimerkeistä. Empiriaosuudessa kuvataan mitä, miten, millä ja missä SVG:tä testattiin. Lisäksi esitettiin testituloksien pohjalta taulukot selaintuesta. Teoriaosuuden ja testituloksien pohjalta johdettiin myös lyhyt ohjeisto SVG:n käytöstä verkkosivulla. SVG:tä käsiteltiin teoria- sekä empiriaosuudessa vain sen perusominaisuuksien osalta. Tutkimuksen ulkopuolelle rajattiin javascript -kielen käyttö. Myös SVG:n animointi, niin javascriptin, CSS:n kuin SMIL -rajapinnankin avulla, rajattiin tutkimuksen ulkopuolelle.

Tähän mennessä projektia oli edistetty oman harrastuneisuuden pohjalta tehdyllä tutkimustyöllä. Myös ympäristö, jossa selaintukea testattiin, oli pystytetty, ja laitteet, joilla testit suoritettiin, valittu. Projektin tavoitteena oli antaa sen tekijälle hyvä tietotaito SVG:n mahdollisuuksista, rajallisuudesta ja käytöstä verkkosivun käyttöliittymäkerroksen rakentamisessa. Tutkimuksessa käytettiin termiä ”verkkosivu“, asian yksinkertaistamiseksi. Tutkimustuloksia voidaan kuitenkin hyödyntää myös monimutkaisemmissa ympäristöissä kuten verkkopalvelun tai verkkosovelluksen kehitystyössä.

2 SVG

SVG on W3C-järjestön kehittämä kuvaformaatti kahdensuuntaisen (2d) grafiikan esittämiseen (W3C 2004). SVG -kuva koostuu XML-elementeistä ja niiden attribuuteista. Elementeillä kuvataan, mistä osista kuva koostuu ja attribuuteilla annetaan elementille lisäohjeita. Koska SVG on XML-dokumentti, voidaan dokumenttia ja samalla kuvaa muokata tekstipohjaisesti. Näin ollen kuvan muokkaukseen ei tarvita erikseen graafisen ohjelmiston vaan esimerkiksi värin muuttaminen kuvassa voidaan tuottaa kooditasolla väriarvoa muuttamalla.

SVG:n ehdottomiin etuihin kuuluu sen pikseliriippumattomuus. Se ei ole riippuvainen laitteen näytön tarkkuudesta, minkä ansiosta se näyttää terävältä laitteen resoluutiosta riippumatta. (Soueidan 2015, Coyer 2016). Se ei menetä terävyyttään suurennettaessa (kuva 1) toisin kuin bittigrafiikka (kuva 2). Bittigrafiikkaan, kuten PNG ja JPG, nähden SVG on monessa muussakin tilanteessa parempi vaihtoehto. Skaalautuvuuden lisäksi SVG:n tiedoston koko on mahdollista pitää pienempänä vastaavaan bittigrafiikkakuvaan verrattuna. (Coyer 2016). SVG on myös esteetön (saavutettava), jonka ansiosta se soveltuu paremmin näkörajoitteisille verkkosivun käyttäjille. Lisäksi SVG:tä voidaan muokata kooditasolla (Coyer 2016, 1). Kuvan muokkausta varten ei siis välttämättä tarvita graafista ohjelmistoa. Tällainen esimerkki voisi olla kuvan värin vaihtaminen, jota varten muutetaan SVG -dokumentista väriarvoa tai vaihtoehtoisesti määritellään se CSS -tyylien avulla.

Kuva 1. Bittigrafiikan kyky säilyttää terävyys on rajallinen (Soueidan 2015)

Kuva 2. SVG kuvan kyky säilyttää terävyys on rajaton (Soueidan 2015)

Vaikka SVG on kuulunut W3C:n kehityskohteisiin vuodesta 1999 (Soueidan 2015), on SVG:n suosio verkkosivuilla jäänyt mielestäni bittigrafiikan varjoon. Selainten kehittyminen ja varsinkin SVG:n tukeminen Internet Explorer versiosta 9 eteenpäin on luultavasti osaltaan johtanut siihen, että SVG -kuvia nähdään verkkosivuilla nyt enenevässä määrin. Samalla SVG on noussut Web -kehittäjäyhteisöissä ajankohtaiseksi puheenaiheeksi yhdessä uusien CSS ominaisuuksien kanssa. Tämä näkyy mielestäni aihetta käsittelevien julkaisujen ja konferenssipuheiden lisääntyvässä määrässä.

2.1 SVG:n versiot

W3C on kehittänyt SVG:stä useita eri versioita ja niiden kirjo saattaa aiheuttaa hämmennystä. W3C:n mukaan SVG -versioiden suosituksia on tällä hetkellä kolme:

1. SVG 1.1 spesifikaation toinen ”painos” (2nd Edition)
2. SVG Tiny 1.2
3. SVG Mobile 1.1

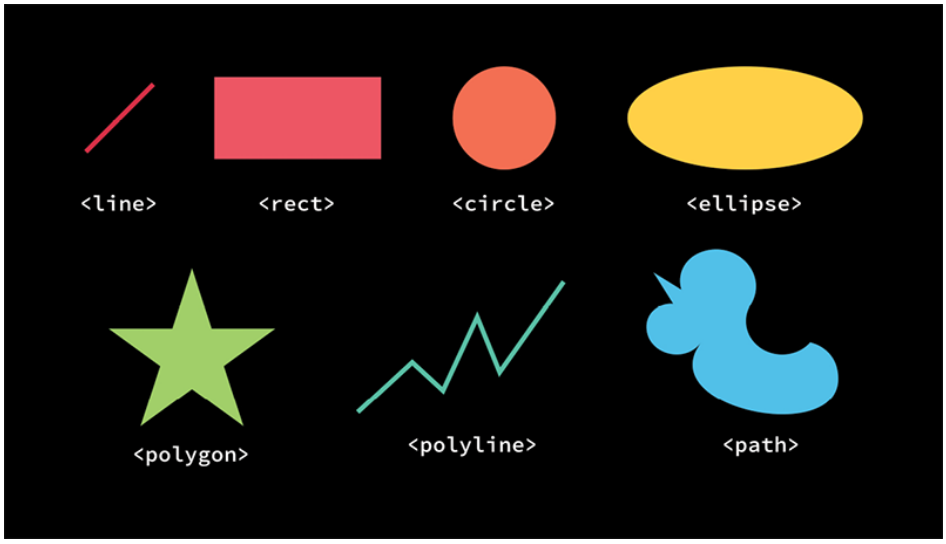
Sekä SVG Tiny 1.2 että SVG Mobile 1.1 ovat ominaisuuksiltaan karsitumpia versioita SVG 1.1 verrattuna. Nämä ominaisuuksiltaan vaatimattomammat version on kehitetty mm. mobiililaitteita varten. (W3C). Nykyisten laitteiden ja selainten tekninen kehitys on kuitenkin sillä tasolla, että SVG 1.1 on se versio, jota käytetään modernissa Web -kehityksessä.

3 SVG:n rakenne

SVG -dokumentti koostuu elementeistä ja niiden attribuuteista. Yksinkertaisimmillaan SVG -kuva on muoto, jolla on muutama ominaisuus. Muoto, kuten neliö, kuvataan elementillä. Neliön ominaissudet, kuten koko ja väri, kuvataan attribuuteilla. SVG -spesifikaatiossa on esitelty yhteensä 80 elementtiä sekä useampi sata attribuuttia (W3C). Keskeisten elementtien ja attribuuttien ansiosta voidaan kuitenkin tuottaa jo monenlaista grafiikkaa ja kuvaa.

3.1 Keskeiset elementit

SVG -elementti (`svg`) on dokumentin hierarkiassa ylimmän tason elementti, jonka jälkeen voidaan vasta kuvata kuvan sisältöä esittäviä elementtejä. Sisältöä esittävistä elementeistä yleisimpiä ovat perusmuodot, murtoviiva ja polku (kuva 3). Mozilla Developer Network (MDN) -sivuston mukaan `svg` kuvaa SVG -kuvan fragmenttia. Se kehystää koko SVG -kuvan ja antaa sille koordinaatiston.



Kuva 3. SVG:n perusmuodot, polyline ja path (Coyer 2016)

Title- ja **desc** -elementtejä käytetään kuvamaan SVG -kuvaa sanallisesti. Nämä elementit voivat sisältää ainoastaan tekstiä ja ne eivät piirry SVG -kuvaan. Sen sijaan ne voivat antaa selaimelle vaihtoehtoisen tavan käsitellä SVG:tä. (W3C, SVG spesifikaatio).

Text on tekstin esittämiseen käytettävä elementti (MDN). Se voi siis sisältää vain tekstiä, mutta toisin kuin `title` ja `desc` -elementtien kohdalla, `text` -elementin sisältö näytetään kuvassa. Käyttäjä pystyy valitsemaan (maalaamaan) elementin sisällön samalla tavalla kuin muunkin verkkosivulla esitettävän tekstisisällön.

G -elementin avulla kerätään useampi SVG:n elementti omaksi ryhmäksi (MDN). Elementtiä käytettäessä on tarpeellista antaa sille id tai/ja luokka, jotta ryhmää voidaan hallita CSS:n avulla. Liikemerkin eri osien piilottaminen voi olla tarpeellista ja perusteltua, kun halutaan tiivistää ja selkeyttää verkkosivun sisältöä mobiililaitteille sopivaksi (kuva 4).

Kuva 4. SVG -kuvan osien piilottamisessa kannattaa hyödyntää `g` -elementtiä (Soueidan 2015).

Defs -elementin avulla kuvataan se osa SVG:stä, joka halutaan näyttää myöhemmin verkkosivun DOM rakenteessa (Coyer 2016, 3).

Symbol -elementtiä käytetään yhdessä `use` -elementin kanssa. Sen avulla esitellään SVG –kuva, joka `use` –elementin avulla näytetään halutussa kohdassa verkkosivua. Kun `symbol` – `use` -yhdistelmää käytetään, `viewBox` siirtyy `symbol` -elementistä `use` -elementtiin. Tämä on erityisen hyödyllinen ominaisuus, kun halutaan esittää esimerkiksi kaikki verkkosivun ikonit samassa SVG -kuvassa ja käyttää niitä myöhemmin yhdessä tai useammassa kohdassa. (Coyer 2016, 3).

Use -elementin avulla kopioidaan aikaisemmin esitelty SVG -dokumentin osa ja liitetään se verkkosivulla haluttuun kohtaan. Se siirtää mukanaan koko dokumentin osan kaikkine elementteineen. (MDN)

Pattern -elementtiä käytetään kuvamaan kuvio, jota toistetaan x- ja y -suunnassa. Se muodostaa ikään kuin visuaalisen tekstuurin, jota käytetään muiden elementtien täytteenä. Kaikki ne elementit, jotka hyödyntävät `fill` -attribuuttia, voivat hyödyntää `pattern` -elementtiä. (MDN). Esimerkiksi neliönmallinen elementti voidaan värin sijasta täyttää kuvioinnilla kuten pienillä ympyröillä (esimerkki 1).

```
<svg … >
    <pattern id="kuvio" x="0" y="0" width="20" height="20"
patternUnits="userSpaceOnUse">
        <circle cx="10" cy="10" r="10" fill="#000"></circle>
    </pattern>
    <rect x="0" y="0" width="100%" height="100%"
fill="url(#kuvio)"></rect>
</svg>
```

Esimerkki 1. Toistuva kuvio (`pattern`) neliön (`rect`) täytteenä

3.2 Keskeiset attribuutit

SVG:n attribuuteilla annetaan elementeille lisäohjeita, kuten minkä värinen tai kokoinen elementin tulee olla. Attribuutit on jaettu SVG -spesifikaatiossa useampaan kategoriaan (W3C) ja ainoastaan esittämiskerroksen attribuutteja (presentation attributes) voidaan hallita myös CSS:n avulla (MDN).

Xlink:href on SVG:n sisällä linkityksiin käytettävä attribuutti. Sen avulla voidaan viitata joko SVG:n sisällä olevaan elementtiin tai SVG kuvan ulkopuolella olevaan tiedostoon.

Fill on keskeisin esittämiskerroksen attribuutti. Sen avulla määritetään SVG -elementtien täyte. Se voi olla väriarvo, viittaus toiseen elementtiin tai tiedostoon.

```
fill=”#323232”
fill=”url(#kuvio)”
fill=”url(kuva.jpg)”
```

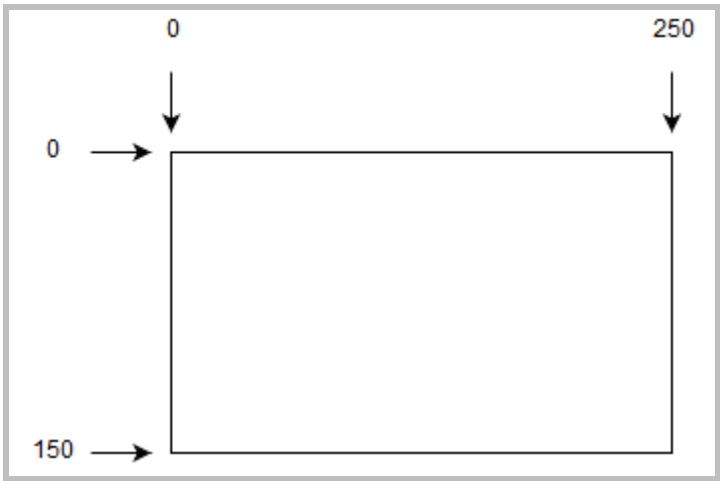
PatternUnits määrittää, mitä koordinaatistoa `pattern` käyttää. Suositeltavaa on, että attribuutin arvona käytetään `userSpaceOnUse`. Tällöin `PatternUnits` käyttää samaa koordinaatistoa kuin `pattern`. (Coyer 2016, 3).

3.2.1 ViewBox

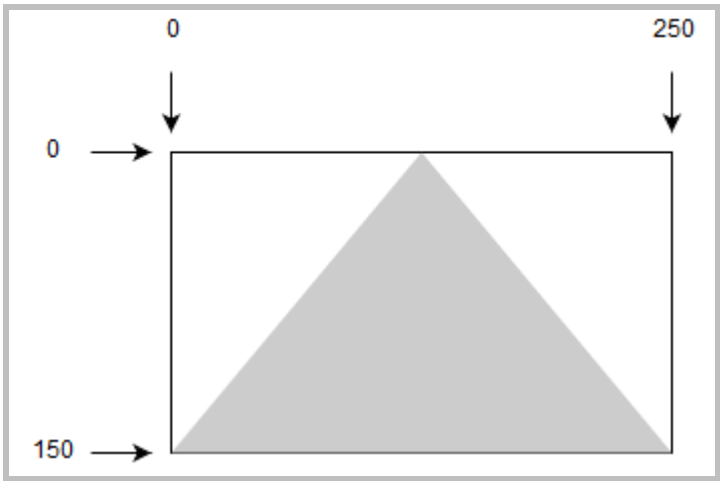
Eräs keskeisimmistä SVG -attribuuteista on `viewBox`, joka määrittää SVG -kuvalle koordinaatiston (kuva 1). Sitä voidaan käyttää useamman eri elementin yhteydessä, mutta tärkein tehtävä sillä on `svg` -elementin yhteydessä, jossa se määrittää minkä kokoiselle alalle grafiikka piirtyy. Toisaalta voidaan myös ajatella, että `viewBox` on ikkuna, jonka läpi

SVG -kuva nähdään (kuva 3). (Soueidan 2014). Attribuutti koostuu neljästä arvosta, jotka ovat:

1. Lähtöpiste vaakasuunnassa
2. Lähtöpiste pystysuunnassa
3. Leveys
4. Korkeus



Kuva 1. ViewBox luo grafiikalle koordinaatiston



Kuva 2. Grafiikka piirtyy koordinaatiston mukaan

3.2.2 PreserveAspectRatio

SVG -spesifikaation mukaan `preserveAspectRatio` on `viewBox` attribuutin lisäominaisuus eikä sillä ei ole vaikutusta, ellei `viewBox` ole määriteltynä (W3C). Attribuutin avulla kerrotaan, kuinka `viewBox` sijoittuu, mikäli `width` ja `height` eivät ole samassa suhteessa kuin `viewBox` (Soueidan 2015). Omien kokemuksieni perusteella se estää kuvaa

vääristymästä ainakin IE11 -mobiiliselaimessa, jossa se oletuksena käyttäytyy kuin `preserveAspectRatio` olisi `none` (kuva 5). Coyer (2016, 6) muistuttaa, että oletuksena SVG käyttäytyy kuin sen `preserveAspectRatio` olisi `xMidYMid meet`.

Kuva 5. SVG -kuva saattaa vääristyä jos `svg`:lle määritellään `preserveAspectRatio="none"` (Coyer 2016, 6)

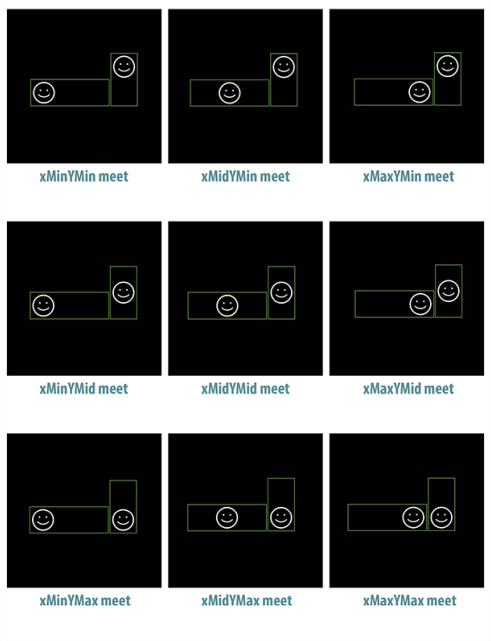
`PreserveAspectRatio`:n ensimmäiset arvot, `x` ja `y`, alustavat missä `viewBox`:n tulee sijaita vaakasuunnassa ja pystysuunnassa. Sekä vaakasuunnassa että pystysuunnassa `viewBox` voidaan sijoittaa kolmeen kohtaan:

- `xMin` – kuva tasataan vasempaan reunaan
- `xMid` – kuva keskitetään vaakasuunnassa
- `xMax` – kuva tasataan oikeaan reunaan
- `yMin` – kuva sijoitetaan yläreunaan
- `yMid` – kuva keskitetään pystysuunnassa
- `yMax` – kuva sijoitetaan alareunaan

Attribuutin viimeinen arvo tulee olla joko `meet` tai `slice`. Jos `viewBox`:lla halutaan täyttää koko `svg`:n pinta-ala, käytetään `slice` -arvoa (kuva 6). Tällöin osa viewBoxista leikkaantuu `svg`:n pinta-alan ulkopuolelle. Mikäli `viewBox`:n halutaan näkyvän kokonaisuudessaan `svg`:n pinta-alalla, käytetään `meet` -arvoa (kuva 7). Huomioitavaa on, että `PreserveAspectRatio`:lla tulee aina määrittää sijainti pystysuunnassa ja vaakasuunnassa, sekä leikataanko kuva vai ei. Sillä ei voi määrittä vain sijaintia vaakasuunnassa, vaikka pystysuunnassa kuvan koko ja koordinaatisto olisivatkin samassa suhteessa. Kuva, jonka ei haluta leikkautuvan ja halutaan olevan keskellä, määritellään seuraavasti:

```
<svg width="…" height="…" viewBox="…"
preserveAspectRatio="xMidyMid meet" …>
  …
</svg>
```

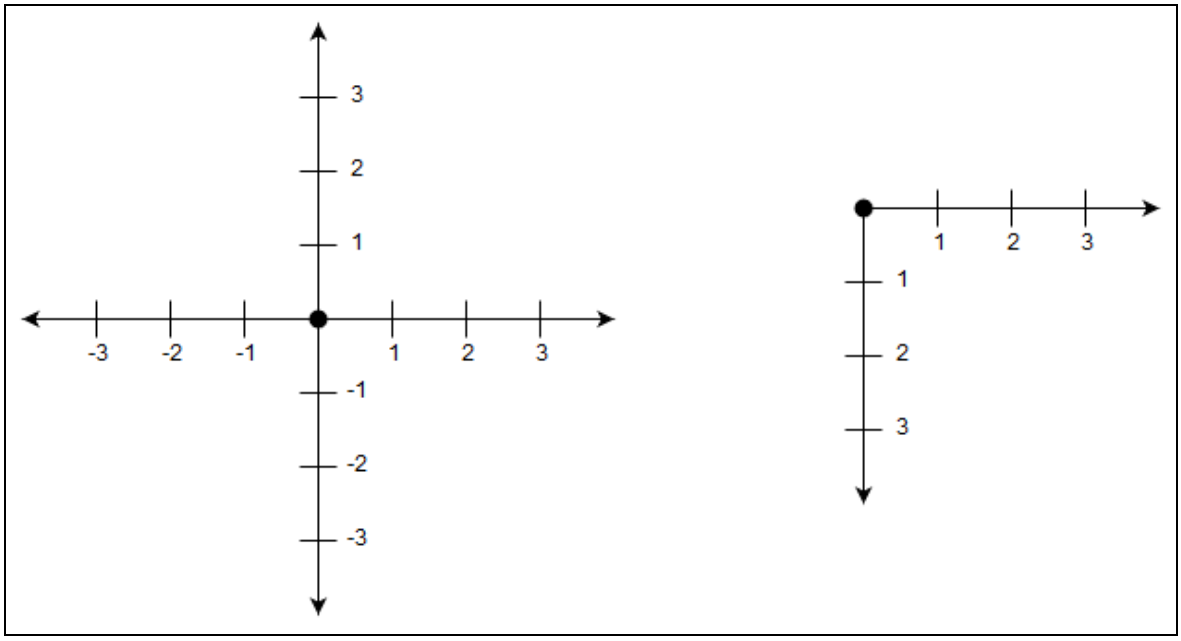
Kuva 6. SVG -kuvan käyttäytyminen `preserveAspectRatio` -attribuutin `slice` -arvolla (Coyer 2016, 6)



Kuva 7. SVG –kuvan käyttäytyminen `preserveAspectRatio` -attribuutin `slice` -arvolla (Coyer 2016, 6)

4 SVG koordinaatisto

SVG:n koordinaatisto eroaa geometrisestä koordinaatistosta siten, että SVG koordinaatistolla ei ole akseleiden leikkauspistettä (kuva 8). SVG:n koordinaatisto alkaa vasemmasta yläkulmasta (0,0) ja jatkaa X -akselia oikealle ja Y -akselia alas. Molemmat akselit saavat ainoastaan positiivisia arvoja. (Soueidan 2014).

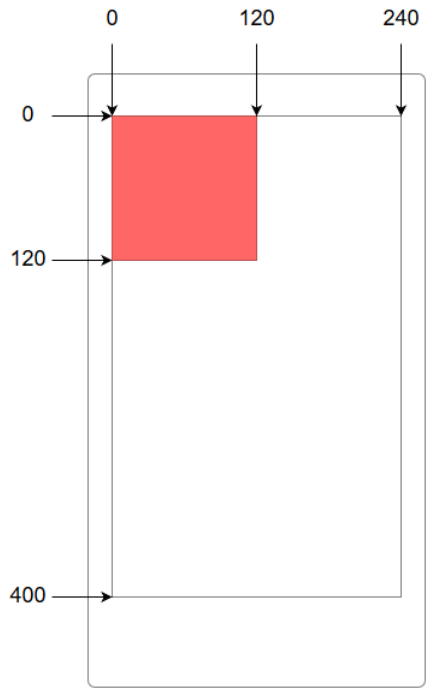


Kuva 8. Geometrinen koordinaatisto vasemmalla ja SVG koordinaatisto oikealla

Verkkosivulla SVG pyrkii oletuksena käyttämään koordinaatistona selainikkunaa, mutta `viewBox` määrittää SVG:lle koordinaatiston, jolle grafiikka piirtyy. Se on SVG:n todellinen koordinaatisto. SVG kuva piirtyy selainikkunalle ilman `width` ja `height` -attribuutteja tai vaihtoehtoisesti ilman `viewBox` -attribuuttia. Selaimen on siis saatava vinkki, minkä kokoiselle alalle kuva piirretään. Mikäli `viewBox` määritellään, niin selain johtaa siitä automaattisesti korkeuden ja leveyden (pikseleinä) vaikka `width` ja `height` -attribuutteja ei erikseen olisi määritelty. Mikäli määritellään `svg`:lle `width` ja `height`, niin selain osaa piirtää kuvan vastaavalle alueelle. Oletuksena selain käyttää pikseleitä, mikäli mittayksikköä ei ole ilmoitettu arvon perässä. Hyväksyttyjä mittayksiköitä ovat; em, ex, px, pt, pc, cm, mm, in, ja prosenntti (%). (Soueidan 2014).

Koordinaatistoa ja sen käyttäytymistä on kuitenkin paljon helpompi ymmärtää kun esitetään esimerkkitapauksia eri tilanteista. Jos `svg`:n `width` ja `height` ovat identtisiä `viewBox` :n kanssa, selain toistaa SVG -kuvan juuri sellaisena kuin odottaa saattaa (kuva 9).

```
<svg width="120" height="120" viewBox="0 0 120 120">
  <rect width="120" height="120" fill="..."/>
</svg>
```

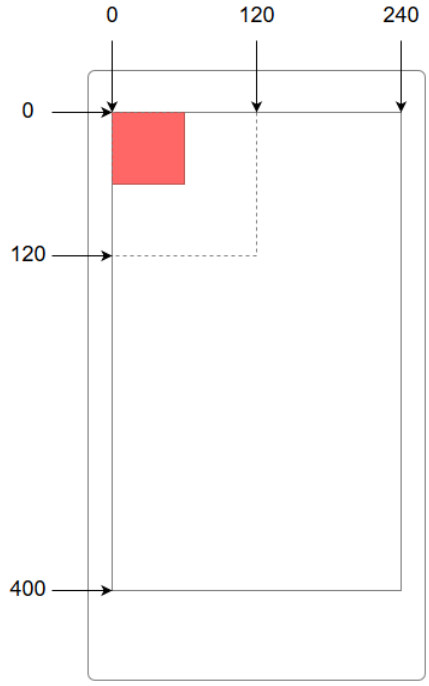


Kuva 9. SVG:n koordinaatisto kun laitteen näytön koko on 240*400 pikseliä ja SVG:n koko 120*120 pikseliä

Mikäli `viewBox` on suurempi kuin sen sisältö, sisällön pinta-ala muuttuu suhteessa koordinaatistoon (esimerkki 2). `Width` ja `height` määrittävät kuitenkin SVG -kuvan varaaman alan (kuva 10).

```
<svg width="120" height="120" viewBox="0 0 240 240">
  <rect width="120" height="120" fill="..."/>
</svg>
```

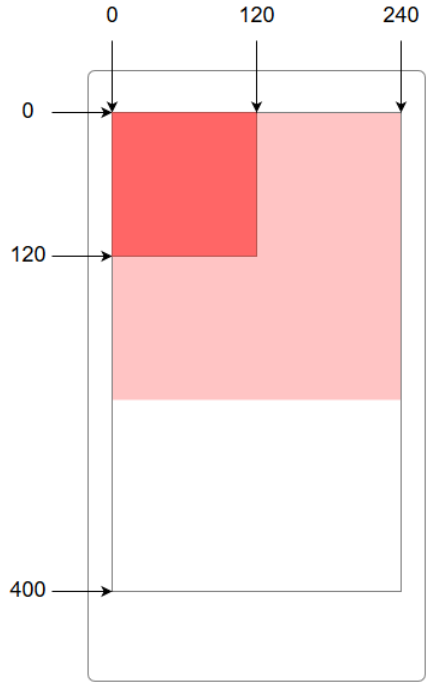
Esimerkki 2. Piirrettävän elementin (`rect`) koko on 120*120 pikseliä, mutta koska koordinaatisto on 240*240, elementti piirtyy näytölle 60*60 pikselin kokoisena.



Kuva 10. SVG:n koordinaatisto kun laitteen näytön koko on 240*400 pikseliä ja `viewBox` on suurempi kuin sen sisällä oleva grafiikka

Mikäli SVG -kuvan sisältö on suurempi kuin `viewBox`, sisältö leikkautuu `viewBoxin` mukaan ja kuvasta näkyy vain osa. Kuvasta näytetään sen verran kuin `viewBox` -attribuutissa määritetään (kuva 11).

```
<svg width="120" height="120" viewBox="0 0 120 120">
  <rect width="240" height="240" fill="..."/>
</svg>
```

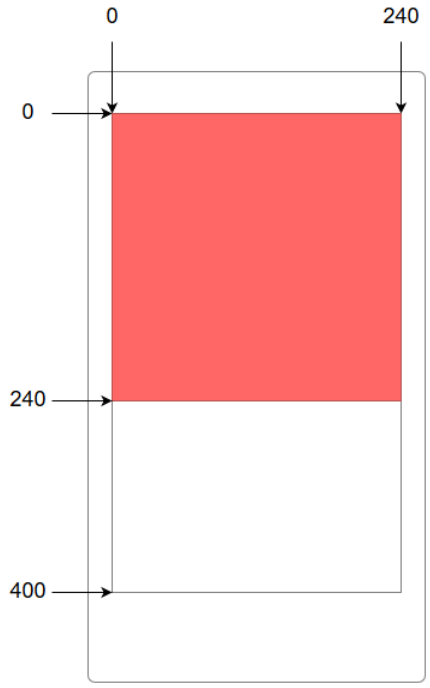


Kuva 11. SVG:n koordinaatisto kun laitteen näytön koko on 240*400 pikseliä ja `viewBox` pienempi kuin grafiikka

.

SVG elementin width ja height attribuuteilla määritetään SVG kuvan absoluuttinen koko selaimessa (kuva 12).

```
<svg width="240" height="240" viewBox="0 0 120 120">
  <rect width="120" height="120" fill="..."/>
</svg>
```



Kuva 12. Viimekädessä `width` ja `height` määräävät, minkäkokoisen alueen SVG verkkosivulta valtaa.

Jotta mahdollisimman moni selain osaa piirtää SVG kuvan halutulla tavalla, täytyy SVG kuvalle määrittää viewBox arvo sekä width ja height arvot. Lisäksi width ja height arvojen on hyvä olla samassa suhteessa viewBox attribuutissa määritetyn korkeuden ja leveyden kanssa. (Soueidan 2015).

5 SVG:n implementointi verkkosivulle

SVG:n yhteydessä on hyvä määritellä versio ja nimiavaruus, jota noudatetaan. Epäselvää on miten tarkasti ne tulee määritellä. MDN:n nimiavaruuteen keskittyvässä artikkelissa on esitetty yksinkertainen pohja nimiavaruuden määrittämiselle.

```
<svg xmlns="http://www.w3.org/2000/svg"
     xmlns:xlink="http://www.w3.org/1999/xlink">
</svg>
```

Soueidanin (2015) ja MDN:n esimerkeissä käytetään myös `version` -attribuuttia. Sen avulla määritellään käytettävä SVG:n versio ja se voi olla vain `1.0` tai `1.1` (W3C). SVG:n implementointi on hyvä aloittaa siis määrittelemällä nimiavaruus ja versio seuraavasti:

```
<svg xmlns="http://www.w3.org/2000/svg"
     xmlns:xlink=http://www.w3.org/1999/xlink
     version=”1.1”
>
</svg>
```

SVG kuvan implementoinnissa verkkosivulle ei ole yhtä tapaa vaan se voidaan liittää verkkosivulle esimerkiksi viittaamalla ulkoiseen tiedostoon tai lisäämällä SVG -dokumentin koodi suoraan HTML:n sekaan. Viittaus voidaan tehdä esimerkiksi `img` -elementtiä käyttämällä. SVG kuva voidaan lisätä verkkosivulle monella eri tapaa, mutta Coyer (2016, 1) esittää kolme yleisintä tapaa jotka ovat:

1. `img` -elementin avulla
2. HTML -elementin taustakuva CSS:n avulla
3. HTML -dokumenttiin upotettu SVG –dokumentti

Coyerin (2016) mukaan helpoin tapa lisätä kuva verkkosivulle on käyttää `img` –elementtiä.

```
<img src=”pallo.svg” alt=”…” />
```

Soueidan (2016) kuitenkin esittää, että tavallisin tapa on käyttää `object` -elementtiä, jonka avulla voidaan esittää vaihtoehto, mikäli selain ei tue SVG:tä. Tällöin `object` –elementin sisäpuolella kuvataan vaihtoehto, joka halutaan näyttää käyttäjälle, jos selain ei tue SVG:tä.

SVG -kuva voidaan liittää verkkosivulle, HTML -elementin taustakuvaksi CSS:n avulla. Tällöin viitataan ulkoiseen SVG –tiedostoon seuraavasti:

```
background-image: url(pallo.svg);
```

5.1 Inline SVG

Yksi tapa implementoida SVG -kuva verkkosivulla on sisällyttää sen koodi suoraan HTML -dokumenttiin (Inline SVG). Tällöin SVG tiedoston sisältö lisätään HTML -dokumentissa haluttuun kohtaan, body -elementin sisälle.

```
<!doctype html>
<html>
<head>
    …
</head>
<body>

    <svg … >
        …
    </svg>

</body>
</html>
```

Coyerin (2016) mukaan Inline SVG kuva antaa monia etuja verrattuna muihin implementointimuotoihin. Näitä etuja ovat:

- CSS -tyylien sitominen SVG -tiedoston elementteihin
- Javascriptin sitominen SVG -tiedoston elementteihin
- Selain ei joudu tekemään erikseen HTTP -pyyntöä kuvaa varten
- SVG -kuvan esteettömyyttä voidaan lisätä ARIA -attribuuttien avulla

Koska SVG on kuvatiedoston lisäksi myös XML -dokumentti, aivan kuten HTML -dokumenttikin, selaimet ymmärtävät hyvin sen syntaksia ja elementtejä. Inline SVG on osa HTML -dokumentin Document Object Model (DOM) –mallia ja sen ansiosta SVG:n

elementteihin voidaan sitoa CSS tyylimäärittelyjä, aivan kuten muihinkin sivuston elementteihin. Inline SVG mahdollistaa myös ARIA -attribuuttien lisäämisen esteettömyyden (saavutettavuus) lisäämiseksi. Suorituskyvyn kannalta koodin sisällyttäminen HTML dokumenttiin vähentää HTTP -pyyntöjen määrää kun selaimen ei tarvitse erikseen hakea SVG -tiedostoa. Toisaalta inline SVG ei voi hyödyntää selaimen välimuistia (cache), joka on myös suorituskykyä parantava tekijä. Selaintuen kannalta Inline SVG on tuettuna varsin hyvin mobiiliselaimissa (taulukko 1).

Taulukko 1. SVG -kuvan tavallisimpien implementointitapojen tuki yleisimmille mobiiliselaimille (Coyer 2016, 1)

	SVG AS <IMG>	SVG AS BACKGROUND-IMAGE	INLINE <SVG>
Any browser on iOS	✔	✔	5.1+
Android Browser (before Chrome became the default Android browser).	3+	3+	3+
Chrome on Android	✔	✔	✔
Firefox on Android	✔	✔	✔
Opera Mini	✔	✔	✘
Opera Mobile	✔	✔	12+
IE Mobile	✔	✔	✔

6 Responsiivinen SVG

Nimestään huolimatta SVG -kuva ei skaalaudu kaikilla verkkosivulla automaattisesti, ellei sitä erikseen määritellä. *Padding-bottom hack* -tekniikkaa käytetään usein iframe -elementtien yhteydessä sivuuttamaan elementin absoluuttinen koko. Hyvän selaintuen saavuttamiseksi myös SVG -kuvalle on asetettava absoluuttinen koko, joka määräytyy width ja height arvojen mukaan. SVG ei siis täytä automaattisesti sen ympärillä olevaa tyhjää tilaa. Bellamy-Royds (2015) muistuttaa että SVG –kuva, jonka `width` on 100%, antaa heikon selaintuen ja esittää responsiivisen SVG -kuvan hallintaan muutamia ratkaisuja joille voidaan taata hyvä selaintuki. Näitä tapoja ovat:

- CSS taustakuva ja padding-bottom (esimerkki 3)
- Inline SVG ja padding-bottom –kehys (esimerkki 4)
- Inline SVG ja inline padding-bottom (esimerkki 5)

CSS

```
.ratio4-3 {
 width: 100%;
 background-image: url(image-with-4-3-aspect-ratio.svg);
 background-size: cover;
 height: 0;
 padding: 0; /* reset */
 padding-bottom: calc(100% * 3 / 4);
}
```

Esimerkki 3. Responsiivinen SVG taustakuvana (Bellamy-Royds 2015)

HTML

```
<div class="scaling-svg-container"
   style="padding-bottom: 92% /* 100% * 55/60 */">
  <svg class="scaling-svg" viewBox="0 0 60 55" >
    <!-- SVG content -->
  </svg>
</div>
```

CSS

```
.scaling-svg-container {
 position: relative;
 height: 0;
 width: 100%;
 padding: 0;
 padding-bottom: 100%;
 /* override this inline for aspect ratio other than square */
}
.scaling-svg {
 position: absolute;
 height: 100%;
 width: 100%;
 left: 0;
 top: 0;
}
```

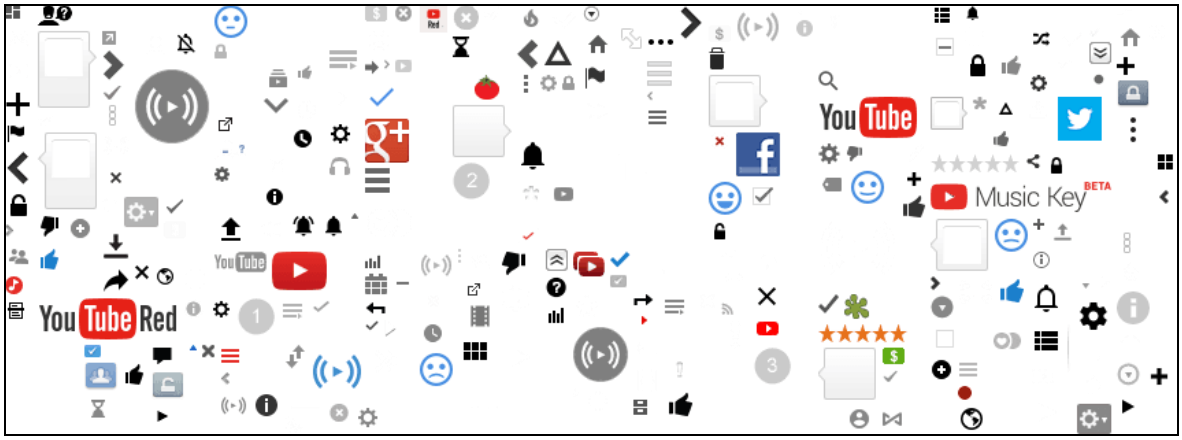
Esimerkki 4.

```
<svg viewBox="0 0 60 55" preserveAspectRatio="xMidYMin slice"
   style="width: 100%; padding-bottom: 92%; height: 1px; over-
flow: visible">
  <!-- SVG content -->
</svg>
```

Esimerkki 5. Responsiivinen inline SVG ilman erillistä kehystä (Bellamy-Royds 2015)

7 SVG sprite

Sprite -tekniikkaa käytetään verkkosivun kuvakkeiden hallintaan ja sillä on verkkosivun suorituskyvyn kannalta positiivisia vaikutuksia. Perinteisesti *sprite* -tekniikkaa on käytetty CSS:n avulla. Ideana on esittää kaikki verkkosivun kuvakkeet yhdessä kuvatiedostossa (kuva 7). Yksittäiselle kuvakkeelle varataan verkkosivulla HTML -elementti, jonka taustakuvaksi kuvatiedosto asetetaan ja kohdistetaan se näyttämään vain haluttu kuvake.

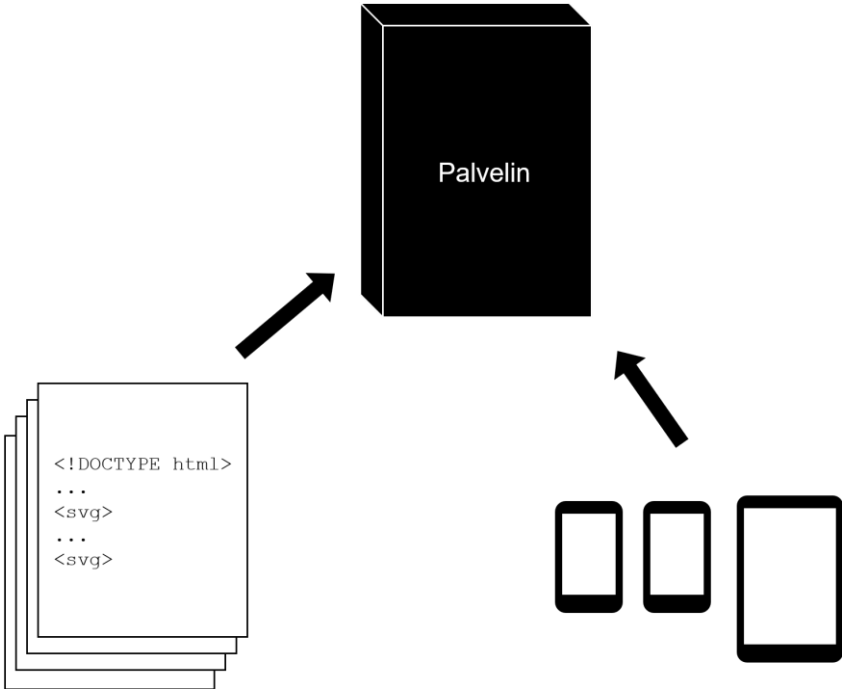


Kuva 13. Youtube.com sivuston kaikki käyttöliittymäkuvakkeet ovat yhdessä PNG -tiedostossa (//s.ytimg.com/yts/imgbin/www-hitchhiker-vfl44vgwb.png)

Upotettu SVG antaa kuitenkin mahdollisuuden käyttää *sprite* –tekniikkaa myös ilman CSS -tyylejä. Tällöin hyödynnetään `sybmol` ja `use` -elemnttejä. Tämä tapa on suorituskyvyn ja ylläpidettävyyden kannalta tehokas, koska samasta kuvakkeesta voidaan myös muokata eri variaatioita CSS:n avulla. (Coyer 2016, 3).

8 Selaintuen testaus

SVG -kuvien selaintuen testauksen suoritin kolmella fyysisellä mobiililaitteella. Kyseisillä laitteilla käytin niiden oletusselaimia, joilla vierailin testitapauksia varten tekemilläni verkkosivuilla. Testilaitteet edustivat kolmea eri käyttöjärjestelmää ja selainta (taulukko 2). Halusin suorittaa testauksen mahdollisimman lähelle tuotantoa muistuttavassa ympäristössä ja sitä varten käytin aiemmin pystyttämääni Apache2 Web -palvelinta, jonne loin jokaiselle testattavalle SVG -kuvalle oman verkkosivun (kuva 13). Verkkosivut sijaitsivat julkisessa internetissä, osoitteessa //mihkellind.com/svg. Kävin jokaisella testilaitteella läpi kaikki verkkosivut ja totesin toistiko selain SVG –kuvan odotetulla tavalla.



Kuva 14. Testiympäristö koostui HTML5 –dokumenteista palvelimesta ja kolmesta testilaitteesta

Taulukko 2. Testilaitteet, niiden käyttöjärjestelmä ja selain

	Merkki ja malli	Käyttöjärjestelmä	Selain
Laite 1	Nokia Lumia 920	Windows Phone 8.1	Internet Explorer Mobile 11
Laite 2	iPad Mini (1st gen.)	iOS 8.3	Mobile Safari 8
Laite 3	OnePlus X	Android 6.0.1	Chrome 54

8.1 SVG ja img

HTML `img` –elementin avulla testattiin yhtä yksinkertaisimmista tavoista liittää SVG verkkosivulle. `Img` elementissä viitattiin ulkoiseen tiedostoon, jonka selain haki HTTP -pyynnöllä . Tässä testissä ulkoinen tiedosto oli kallellaan olevaa kolmiota esittävä kuva, jonka väri oli musta (liite 1). Kaikki testattavat selaimet tukivat tätä implementointitapaa (Taulukko 3).

Taulukko 3. Selaintuki, kun lisätään SVG verkkosivulle img -elementin avulla.

SELAIN	TUKI
Internet Explorer Mobile 11	Kyllä
Mobile Safari 8	Kyllä
Chrome 54	Kyllä

8.2 SVG ja object

SVG:n käyttö `object` –elementin avulla tuotti kaikissa selaimissa odotetun tuloksen (liite 2). Testitulosten perusteella kaikki testattavat selaimet tukivat myös tätä implementointitapaa ongelmitta (Taulukko 5).

Taulukko 4. Selaintuki, kun lisätään SVG verkkosivulle `object` -elementin avulla.

SELAIN	TUKI
Internet Explorer Mobile 11	Kyllä
Mobile Safari 8	Kyllä
Chrome 54	Kyllä

8.3 SVG ja CSS background

SVG HTML -elementin taustakuvana (Liite 3) tuotti kaikissa selaimissa odotetunlaisen tuloksen (taulukko 6).

Taulukko 5. Selaintuki, kun lisätään SVG taustakuvaksi CSS:n avulla.

SELAIN	TUKI
Internet Explorer Mobile 11	Kyllä
Mobile Safari 8	Kyllä
Chrome 54	Kyllä

8.4 SVG:n pattern

SVG:n pattern –ominaisuudessa testattiin tukea inline SVG tukea ja `pattern` -elementin tukea (liite 4). Kaikki selaimest toistivat kuvan odotetulla tavalla (taulukko 7).

Taulukko 6. SVG:n pattern selaintuki

SELAIN	TUKI
Internet Explorer Mobile 11	Kyllä
iOS Safari	Kyllä
Chrome for Android 54	Kyllä

8.5 Responsiivinen SVG

Responsiivinen SVG on kuva, joka täyttää automaattisesti tyhjän tilan sille. Tätä varten testasin Bellamy-Roydsin (2015) mainitsemat tekniikan, jolla oli tiedettävästi hyvä selaintuki. Sen avulla inline SVG:lle annetaan style attribuutin, jossa määritellään CSS:n avulla SVG:n `width` prosentteina. Sitä vastoin `height` tulee olla 1px, mutta `padding-bottom` taas prosentteina suhteessa leveyteen. Viimeiseksi määritin ohjeen mukaan

Taulukko 7. Selaintuki, responsiiviselle SVG -kuvalle

SELAIN	TUKI
Internet Explorer Mobile 11	Kyllä
iOS Safari	Kyllä
Chrome 54 (Android)	Kyllä

8.6 SVG sprite

SVG sprite -kuva testattiin ensisijaisesti sen itsensä takia. Samalla saatiin selville `sym-bol-` ja `use` -elementtien selaintuki (Liite 6). Kaikki testattavat selaimet tukivat SVG:n *sprite* -ominaisuutta (taulukko 9).

Taulukko 8. Selaintuki, SVG -sprite tekniikalle.

SELAIN	TUKI
Internet Explorer Mobile 11	Kyllä
iOS Safari	Kyllä
Chrome for Android 54	Kyllä

9 Pohdinta

Vaikka SVG:n selaintuki ei ole täysin aukoton on sen yleisimpien ominaisuuksien perusteella varteenotettava vaihtoehto verkkosivun grafiikan käytölle. Selaintuen kannalta ei ole syytä jättää SVG:tä pois, mikäli halutaan tukea yleisimpiä mobiiliselaimia. Jos taas selaintukea haetaan vanhemmille IE versioille, on syytä harkita käytetäänkö SVG:tä yhdessä fallback -menetelmän kanssa, vai turvaudutaanko kokonaan bittigrafiikkaan

Tämän työn perusteella SVG tulee käyttää kun halutaan tuottaa laadukasta grafiikkaa verkkosivulle näytön tarkkuudesta ja koosta riippumatta. SVG:n käytössä tulee kuitenkin muistaa ne seikat, jotka tekevät siitä joissain tapauksissa vain vaihtoehdon bittigrafiikalle. Se ei sovellu kaikkiin projekteihin sillä sen työstäminen vaatii niin grafiikan teko kuin implementointi vaiheessa selkeästi enemmän työtä verrattuna bittigrafiikkaan. Mikäli kuitenkin verkkosivun pääpaino on datan visualisoinnissa, voidaan SVG:n ehdottomaksi eduksi laskea sen kyky tarjota interaktiivista grafiikkaan. Vaikka testatuissa SVG kuvissa käytettiin äärimmäisen yksinkertaisia kuvia, koostuivat ne elementeistä, joista voidaan rakentaa myös monimutkaisempia kuvia. Kartat, diagrammit, ikonit, liikemerkit soveltuvat SVG:n käyttöön.

Suorituskyvyn ja sitä kautta käyttökokemuksen parantaminen ovat avainasemassa kun puhutaan kuvien käytöstä verkkosivulla. SVG voidaan nähdä sielultaan täysin laiteriippumattomana kuvamuotona. Sen käytöstä ei kärsi mobiilikäyttäjä eikä se myöskään tuota huonompaa jälkeä suurella näytöllä. Bittigrafiikan käytössä joudutaan aina tilanteeseen, jossa valitaan mitä laitekategoriaa ensisijaisesti palvellaan. Pieni kuva ja pieni tiedostokoko palvelevat parhaiten pieniä näyttöjä, joiden näytön tarkkuus ei ole korkea. Suuresta kuvasta (suuri tiedostokoko) hyötyvät käyttäjät, joilla on hyvin tarkka ja/tai iso näyttö se nopea internetyhteys.

Tutkimuksen tuloksena syntyneitä verkkosivuja voi hyödyntää kuka tahansa. Ne ovat julkisessa verkossa nähtävillä ja esimerkiksi selaintukeen keskittyvässä laitelaboratoriossa voidaan testata laajempi selaintuki tässä työssä testatuille SVG -kuville.

Lähteet

Bellamy-Royds, A. 6.1.2015. How to Scale SVG. Luettavissa: https://css-tricks.com/scale-svg. Luettu: 20.11.2015.

Coyer, C. 2016. Practical SVG, A Book Apart

Friedman, V., Mall, D., Callahan, B., Webb, E., Soueidan, S., Gillenwater, Z M., Stein, B., Weiss, Y., Carneiro, F., Maslen, T., Clarke, A., Allsopp, J., Gaunt, M. 2015. Real-Life Responsive Web Design. Smashing Magazine GmbH. Freiburg, Saksa

Mozilla Developer Network (SVG element reference − SVG). Luettu: 1.11.2016. Luettavissa:
https://developer.mozilla.org/en-US/docs/Web/SVG/Element/svg

Mozilla Developer Network (Namespaces Crash Course). Luettu 7.11.2016. Luettavissa:
https://developer.mozilla.org/en-US/docs/Web/SVG/Namespaces_Crash_Course

Scalable Vector Graphics (SVG) 1.1 (Second Edition) W3C Recommendation 16 August 2011. Luettavissa:
http://www.w3.org/TR/2011/REC-SVG11-20110816/single-page.html. Luettu: 21.10.2016.

Understanding SVG Coordinate Systems and Transformations (Part 1) — The viewport, viewBox, and preserveAspectRatio. Luettavissa:
https://sarasoueidan.com/blog/svg-coordinate-systems/. Luettu 14.2.2016.

W3C – About SVG. 29.10.2004
Luettavissa: https://www.w3.org/Graphics/SVG/About.html
Luettu: 1.12.2016

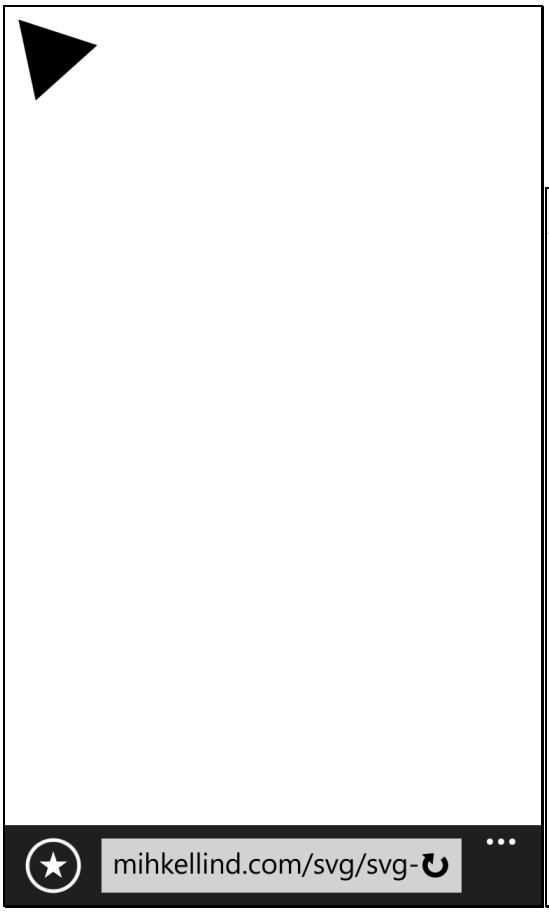
Liitteet

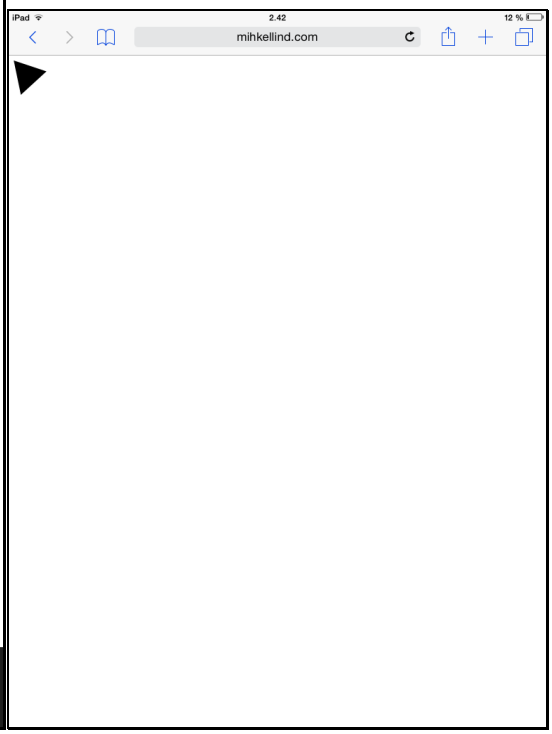
Liite 1. HTML -dokumentti ja kuvakaappaukset – SVG ja img

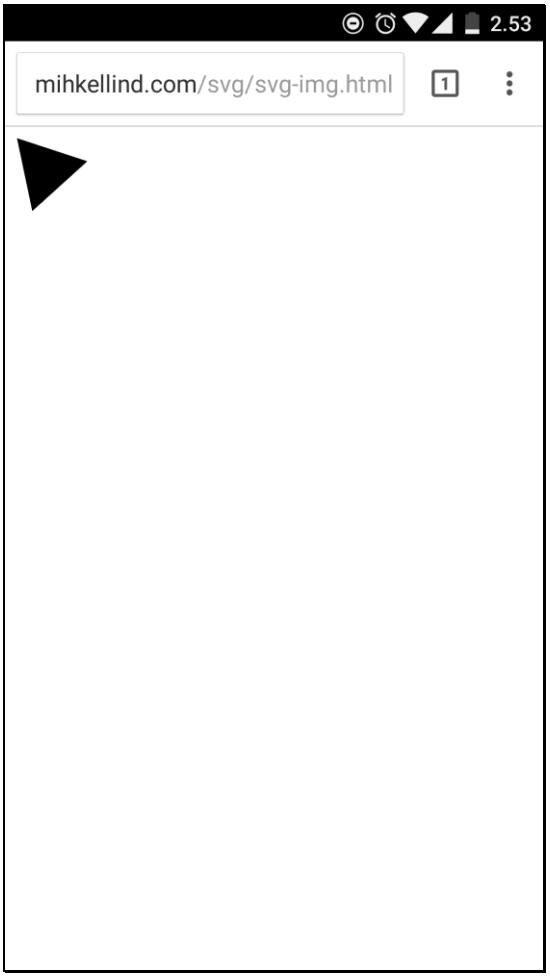
```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-
scale=1">
    <title>SVG img</title>
</head>
<body>

    <img src="kolmio.svg" alt="kolmio">

</body>
</html>
```





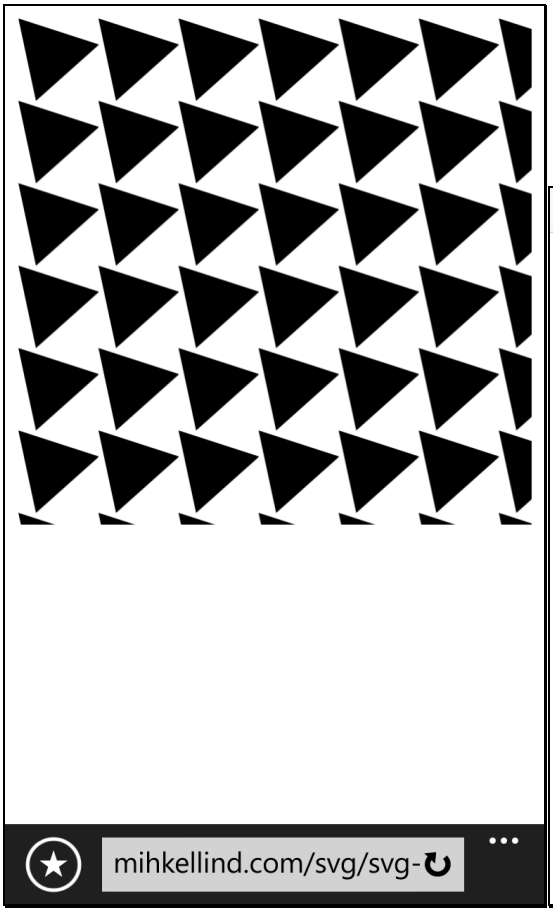

2.53
mihkellind.com/svg/svg-img.html
1

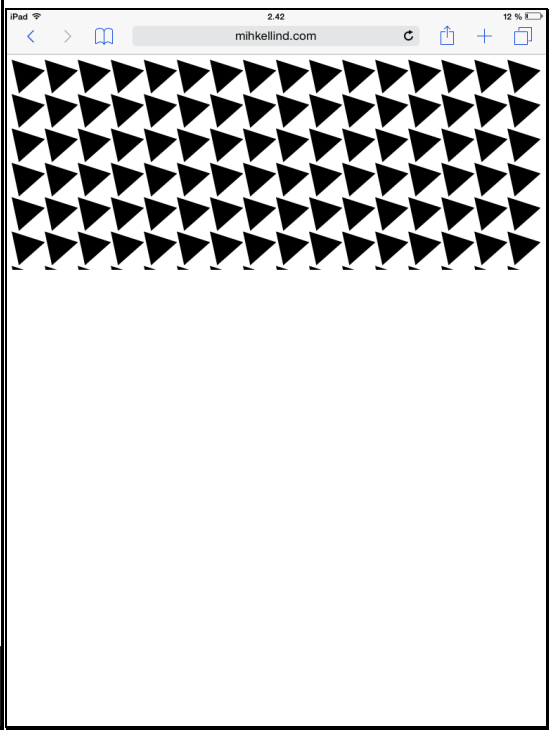
Liite 2. HTML -dokumentti ja kuvakaappaukset – SVG ja CSS background

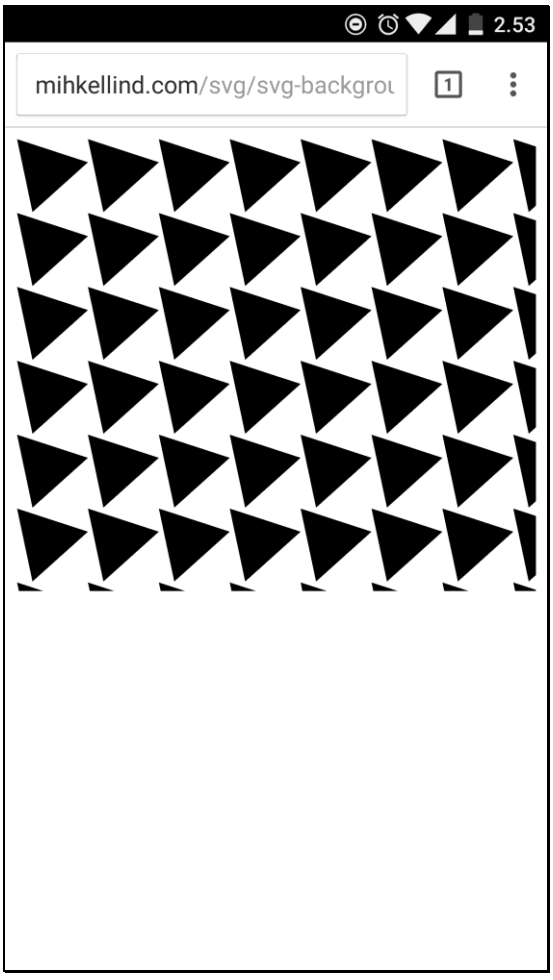
```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-
scale=1">
    <title>SVG background</title>
    <style>
        div {
            height: 300px;
            width: 100%;
            background-image: url(kolmio.svg);
            background-repeat: repeat;
        }
   </style>
</head>
<body>

    <div></div>

</body>
</html>
```


mihkellind.com/svg/svg-


mihkellind.com

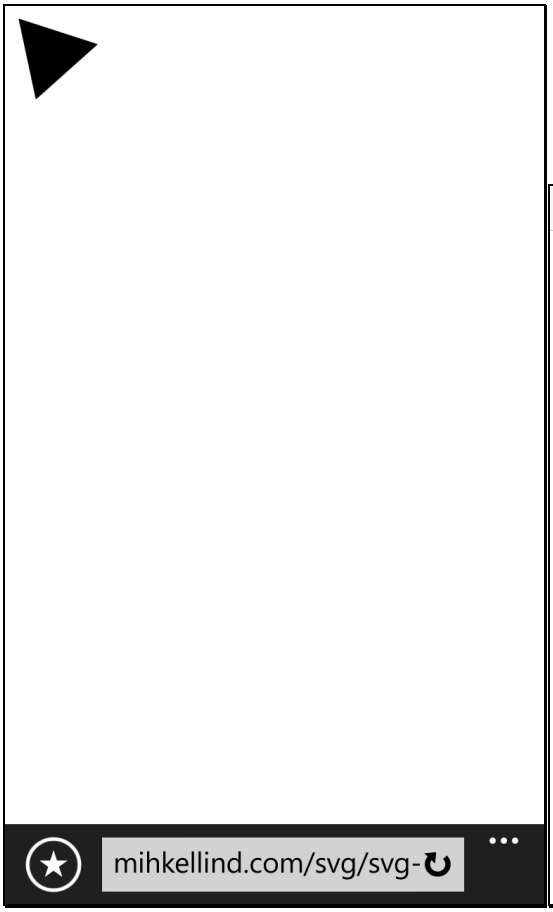

2.53
mihkellind.com/svg/svg-backgrou

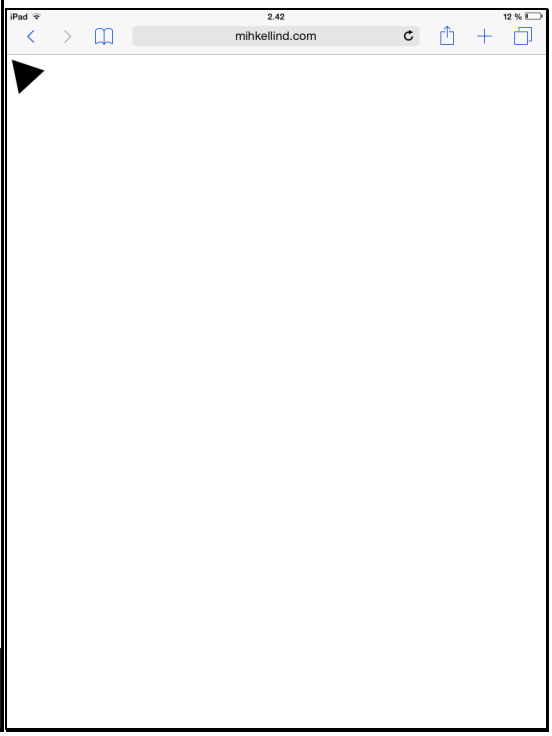
Liite 3. HTML -dokumentti ja kuvakaappaukset – SVG ja object

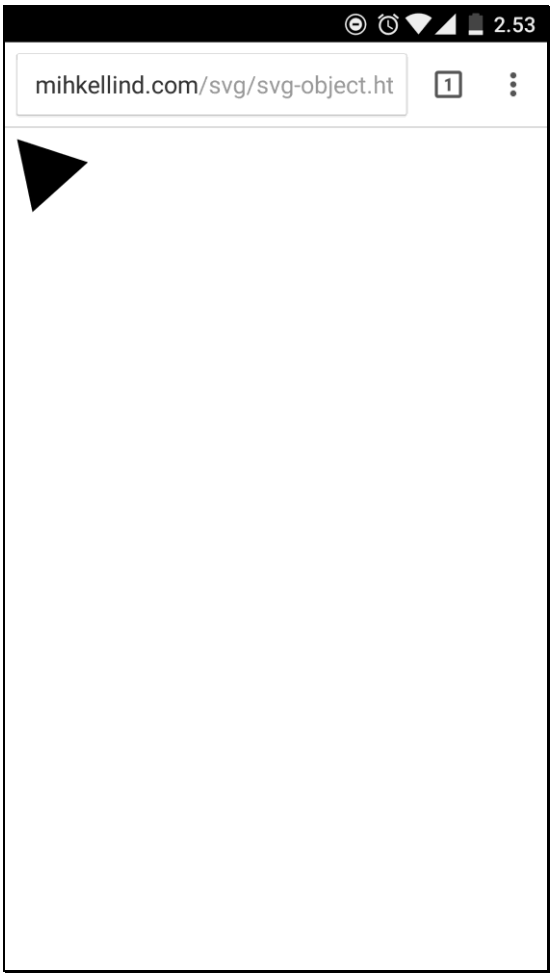
```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-
scale=1">
    <title>SVG Object</title>
</head>
<body>

    <object data="kolmio.svg" type="image/svg+xml">
        <p>Tämä selain ei tue SVG:tä</p>
    </object>

</body>
</html>
```


mihkellind.com/svg/svg-


iPad
2.42
12 %
mihkellind.com

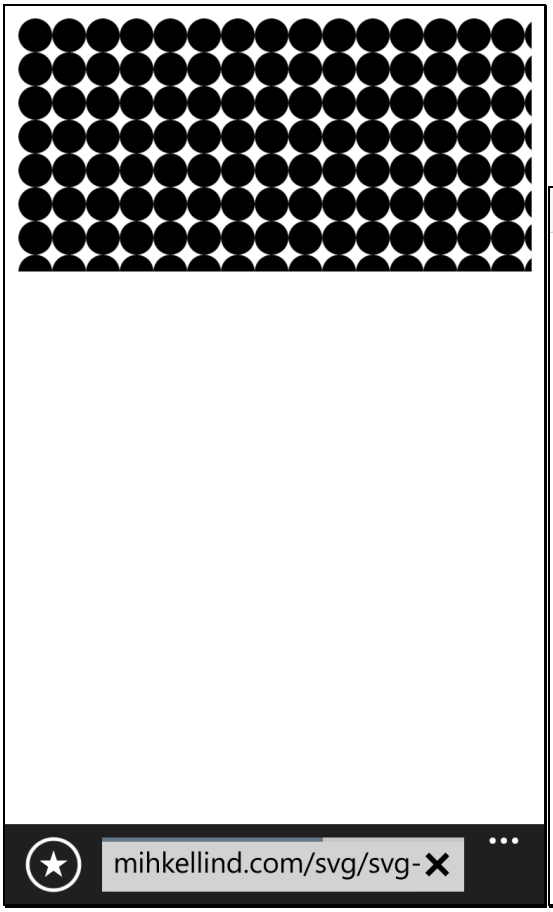

2.53
mihkellind.com/svg/svg-object.ht
1

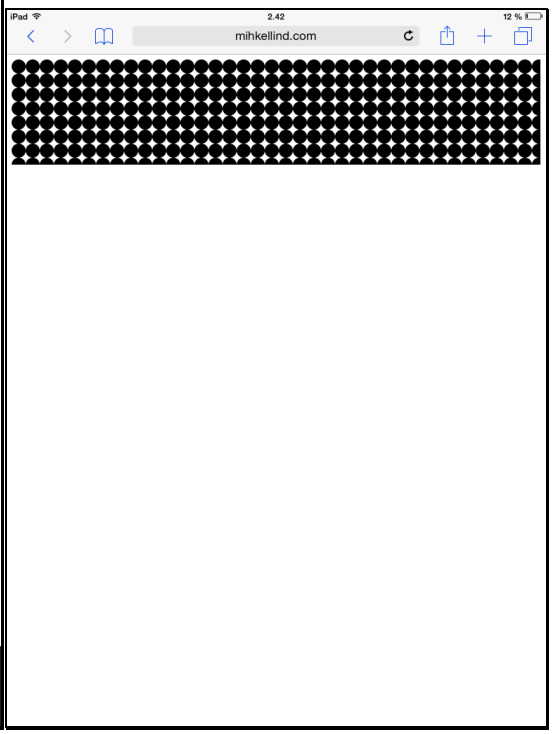
Liite 4. HTML -dokumentti ja kuvakaappaukset – SVG pattern

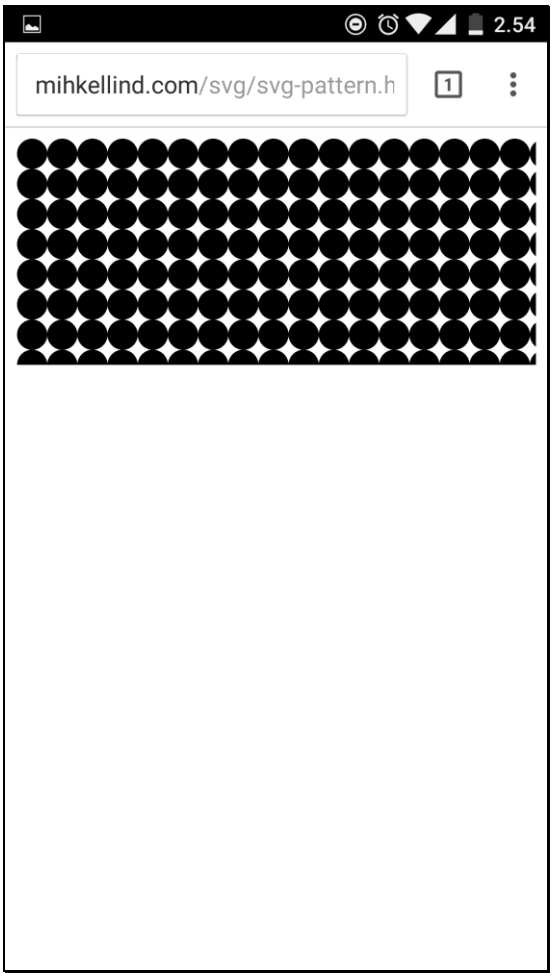
```
<!DOCTYPE html>
<html lang="en">
<head>
   <meta charset="UTF-8">
   <meta name="viewport" content="width=device-width, initial-
scale=1">
   <title>SVG Pattern</title>
</head>
<body>

<svg  width="100%" height="100%" >
    <pattern id="toistuva-kuvio" x="0" y="0" width="20"
height="20" patternUnits="userSpaceOnUse">
        <circle cx="10" cy="10" r="10" fill= "#000" />
    </pattern>
    <rect x="0" y="0" width="100%" height="100%"
fill="url(#toistuva-kuvio)" />
</svg>

</body>
</html>
```

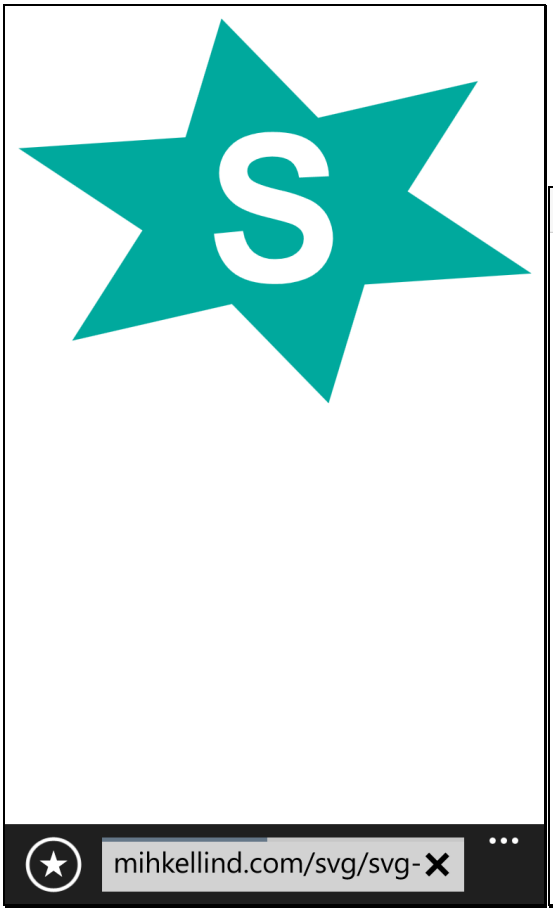

mihkellind.com/svg/svg-

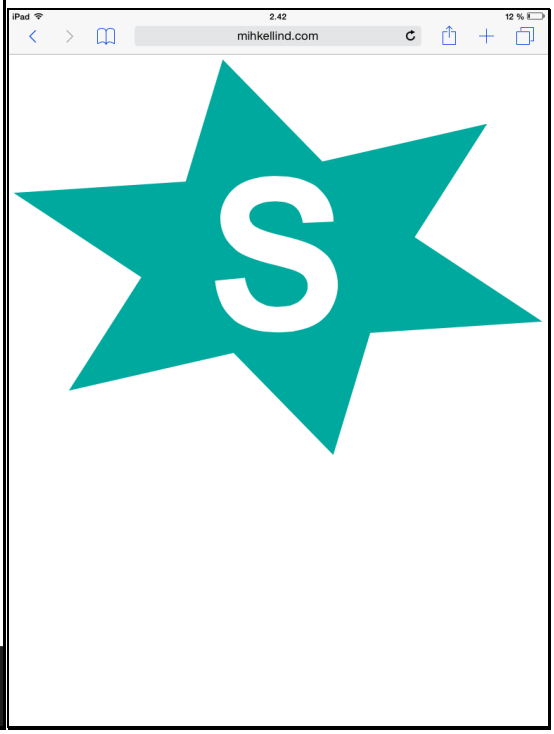

iPad
2.42
12 %
mihkellind.com

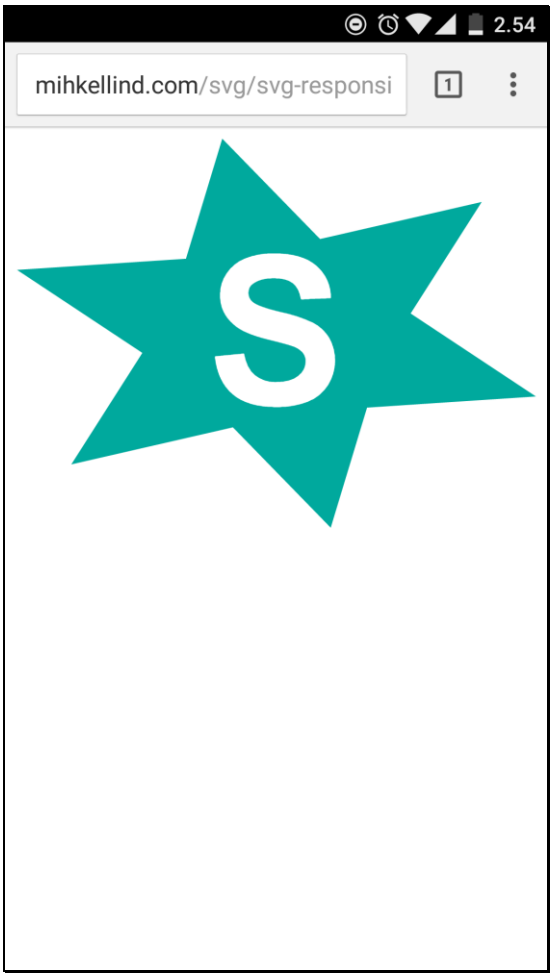

2.54
mihkellind.com/svg/svg-pattern.h
1

Liite 5. HTML .dokumentti ja kuvakaappaukset – Responsiivinen SVG

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-
scale=1">
    <title>Responsiivinen SVG</title>
</head>
<body>
<svg xmlns="http://www.w3.org/2000/svg" version="1.1" viewBox="0
0 200 150" preserveAspectRatio="xMidYMin slice"
    style="width: 100%; padding-bottom: 75%; height: 1px; over-
flow: visible">
    <polygon
            points="200 99.4 134.9 103.7 120.9 150 83.2 111.3
20.9 125.6 48.3 82.6 0 50.6 65.1 46.3 79.1 0 116.8 38.7 179.1
24.4 151.7 67.4 200 99.4"
            fill="#00a99d"/>
    <path d="M134,159.5l11.3-
1.1q1,5.7,4.1,8.3a12.5,12.5,0,0,0,8.4,2.7q5.6,0,8.4-
2.4a7,7,0,0,0,2.8-5.5,5.2,5.2,0,0,0-1.2-3.5,9.5,9.5,0,0,0-4.2-
2.5q-2-.7-9.3-2.5-9.3-2.3-13-5.7a14.9,14.9,0,0,1-5.3-
11.5,14.7,14.7,0,0,1,2.5-8.2,15.5,15.5,0,0,1,7.1-
5.8,28.6,28.6,0,0,1,11.3-
2q10.8,0,16.2,4.7a16.7,16.7,0,0,1,5.7,12.6l-11.6.5q-0.7-4.4-3.2-
6.3t-7.3-1.9q-5,0-7.9,2.1a4.2,4.2,0,0,0-
1.8,3.6,4.4,4.4,0,0,0,1.7,3.5q2.2,1.8,10.6,3.8a56.1,56.1,0,0,1,1
2.5,4.1,16.1,16.1,0,0,1,6.3,5.8,18,18,0,0,1-
.4,18.3,16.4,16.4,0,0,1-7.7,6.4,32.5,32.5,0,0,1-12.5,2.1q-
10.9,0-16.7-5T134,159.5Z"
          transform="translate(-57.8 -75.6)" fill="#ffffff"/>
</svg>
</body>
</html>
```


S
mihkellind.com/svg/svg-

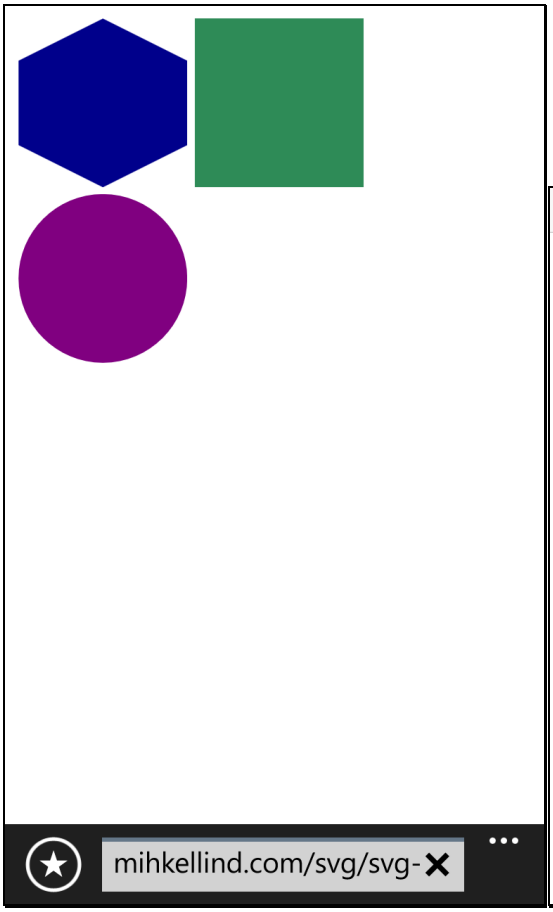

iPad
2.42
12 %
mihkellind.com
S

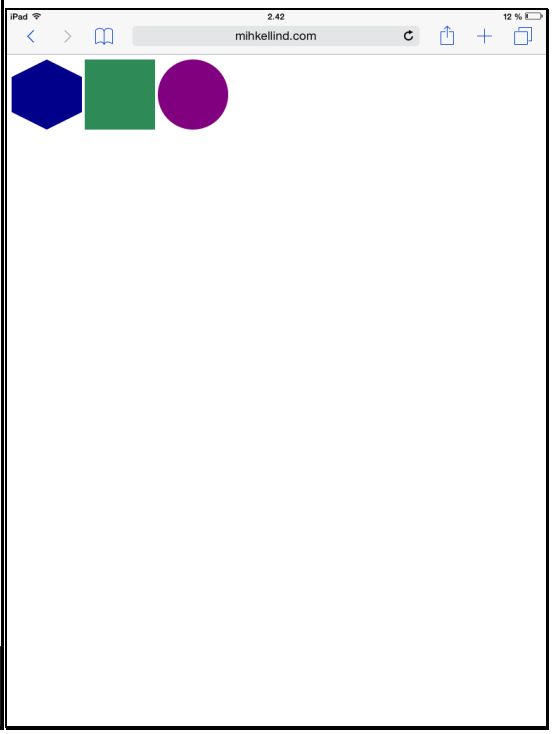

2.54
mihkellind.com/svg/svg-responsi
1
S

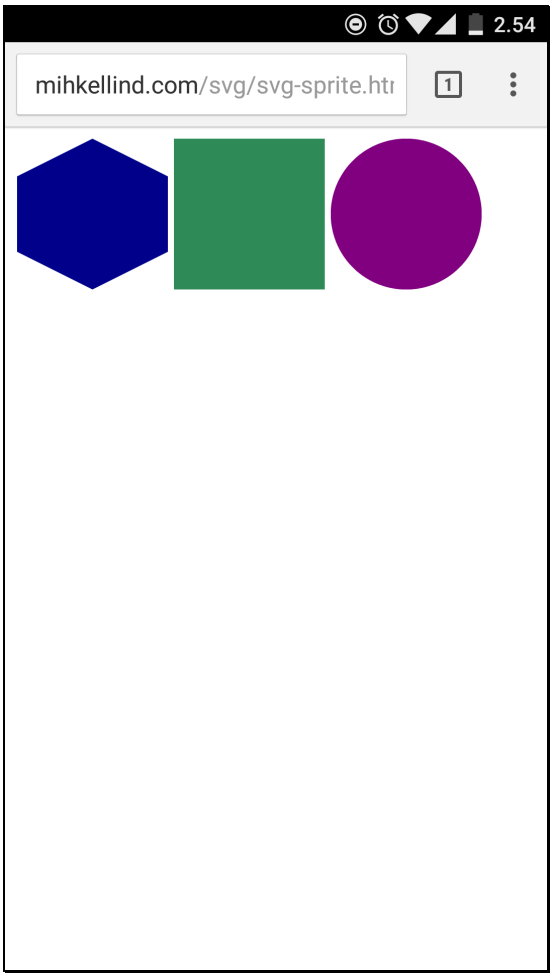
Liite 6. HTML -dokumentti ja kuvakaappaukset – SVG sprite

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <meta charset="UTF-8">
    <title>SVG Sprite</title>
    <style>
        .visually-hidden{
         display: none;
        }
    </style>
</head>
<body>
<svg xmlns="http://www.w3.org/2000/svg" version="1.1" width="0" height="0"
class="visually-hidden">
    <symbol id="rect" viewBox="0 0 100 100">
        <rect width="100" height="100" fill="seagreen"/>
    </symbol>
    <symbol id="circle" viewBox="0 0 100 100">
        <circle cx="50" cy="50" r="50" fill="purple"/>
    </symbol>
    <symbol id="polygon" viewBox="0 0 100 100">
        <polygon points="50 0 100 25 100 75 50 100 0 75 0 25 50 0"
fill="darkblue"/>
    </symbol>
</svg>


<svg width="100" height="100">
    <use xlink:href="#polygon"/>
</svg>
<svg  width="100" height="100">
    <use xlink:href="#rect"/>
</svg>
<svg width="100" height="100">
    <use xlink:href="#circle"/>
</svg>
</body>
</html>
```


mihkellind.com/svg/svg-


mihkellind.com


2.54
mihkellind.com/svg/svg-sprite.htı

Liite 7. Tiivis ohjeisto SVG:n käytöstä verkkosivulla

- Määritä `svg` -elementille aina `width` ja `height`, jotka ovat samassa suhteessa kuin `viewBox`.

- Jos `width` ja `height` eivät ole samassa suhteessa kuin `viewBox`, käytä apuna `preserveAspectRatio`:ta.

- Jos mahdollista, käytä upotettua SVG:tä. Sen avulla SVG:stä tulee osa DOM -rakennetta ja siihen voidaan lisätä vuoropuhelua ja CSS -tyylejä.

–

- CSS:n avulla voidaan hallita vain SVG:n esittämiskerroksen attribuutteja.

- Responsiivinen SVG saavutetaan bottom-padding -tekniikalla. Sillä saavutetaan paras mahdollinen selaintuki responsiiviselle SVG:lle.

- Jos tuotat SVG:n Adobe Illustrator -ohjelmalla, muista tallentaa se Web -käyttöön valitsemalla *Export ja kopioimalla ohjelman generoima koodi*. *Save as* saattaa tuottaa kuvaan tietoa, jota selaimet eivät tue.

- Käytä yksinkertaisia muotoja äläkä tee liian siroja elementtejä. Vaikka SVG pysyy aina tarkkana, eivät kaikki laitteet pysty toistamaan pienimpiä yksityiskohtia.

–

- Piirrä kuva lähtokohtaisesti siinä asennossa jossa se esitetään. Vältä kuvan tai sen osien kääntämistä Adobe Illustratorissa, jotta koodi pysyisi luettavampana.