Silja Karesto

Säännöllisesti toistuvan kuvion toteutus vektorigrafiikkaohjelmalla ja ohjelmoimalla

Metropolia Ammattikorkeakoulu
Medianomi
Viestinnän koulutusohjelma
Opinnäytetyö
27.1.2012

Metropolia

Tiivistelmä

Tekijä(t) Otsikko Sivumäärä Aika	Silja Karesto Säännöllisesti toistuvan kuvion toteuttaminen vektorigrafiikkaohjelmalla ja ohjelmoimalla 57 sivua + 1 liitettä 27.1.2012
Tutkinto	Medianomi
Koulutusohjelma	Viestintä
Suuntautumisvaihtoehto	Digitaalinen viestintä
Ohjaaja(t)	Yliopettaja Pauli Laine

Opinnäytetyön aiheena on tutkia toistuvia kuvioita. Työ on toiminnallinen ja siinä toteutetaan kahdella valitulla medialla säännöllisesti toistuva kuvio. Työllä ei ole toimeksiantoa vaan se tehdään tekijän kiinnostuksesta aiheeseen.

Työn teoriaosuudessa selvitetään lähdekirjallisuuden avulla jaksollisesti ja saumattomasti toistuvien kuvioiden toistumissääntöjä ja elementtien muodostamista. Siinä käsitellään myös matemaattista taustaa pääasiassa jaksottomasti toistuvien kuvioiden osalta. Toiminnallisessa osassa asetetaan tavoitteet ja teoriaosaa hyödyntämällä suunnitellaan säännöllisesti toistuva kuvio. Suunniteltu kuvio toteutetaan käyttämällä Adobe Illustrator vektorigrafiikkaohjelmaa ja Processing-ohjelmointikieltä. Molempien toteutukset selostetaan vaihe vaiheelta. Toteutuksia verrataan keskenään ja todetaan valittujen menetelmien hyödyt ja haitat. Lopuksi tehdään kuvio vielä kerran yhdistämällä Illustrator ja Processing.

Tuloksena oli toistuva kuvio, joka täytti tavoitteet säännöllisestä ja saumattomasta toistuvuudesta. Teoria kuvioiden suunnittelun taustalla oli tarpeen tietää. Käytetyt mediat osoittautuivat yhtä tehokkaiksi työssä suunnitellun kuvion toteutukseen. Kuitenkin muunlaisessa toteutuksessa, esimerkiksi yksityiskohtaisemmassa, niiden erot nousisivat enemmän esiin, joten oli aiheellista pohtia niiden yhdistämistä parhaan lopputuloksen saamiseksi. Ohjelmoinnin ja visuaalisen materiaalin tuottaminen on harvoin yhdistetty toisiinsa, ja tässä työssä sen toimivuus tuotiin esiin.

Avainsanat	toistuva kuvio, ohjelmointi, vektorigrafiikka

Metropolia

Abstract

Author(s) Title Number of Pages Date	Silja Karesto Creating a periodically repeating pattern by vector graphics and programming 57 pages + 1 appendices 27 January 2012
Degree	Bachelor of Culture and Arts
Degree Programme	Degree Programme in Media
Specialisation option	Digital Media
Instructor(s)	Pauli Laine, Senior Lecturer

The aim of this thesis was to examine the theory behind periodically repeating patterns and to create a pattern using the information learned in the theory. The pattern was to be created by using two different media. First using Adobe Illustrator, a vector graphics program, and then a programming language called Processing.

In the theory part different methods of repeating a pattern were explained. Also mathematics behind patterns and their harmony and symmetry were approached. Aims for the pattern were established and the theory was applied to the design of the pattern. Next the process of creating the pattern with the chosen media was followed step by step. The results were then analyzed and compared. After this the two media were combined and the pattern was created one more time.

The results demonstrate the effectiveness of the chosen media in creating a pattern. Knowledge of the theory was proven necessary in the process of designing the pattern. The result is a periodically repeating pattern that fulfills the aims established at the beginning of the process.

In creating this particular pattern the chosen media were equally effective. That however applied only to the type of pattern created in this case. It is concluded that combining the two media proved most effective in the case of more detailed patterns as they both have their pros and cons.

Keywords	repeat patterns, vector graphics, programming, Processing

Sisällys

Liitteet

Liite 1. Valmiin kuvion Processing koodi

1 Johdanto

Työni käsittelee kuvioita, jotka toistuvat saumattomasti ja säännöllisesti sillä ne ovat kiehtoneet minua pidemmän aikaa ja olen myös halunnut toteuttaa sellaisen. Tämän työn puitteissa suunnittelen itse tällaisen kuvion ja opettelen teoriaa sen taustalla. Toteutan kuvion minulle entuudestaan tutulla ohjelmalla, josta minulla on karkea kuva, miten se tehdään. Otan kuitenkin rinnalle myös median, jota en ollut ennen käyttänyt, jotta saan siitä myös enemmän uutta irti. Käyttämällä kahta keskenään hyvin erilaista mediaa joudun harkitsemaan lähestymistapaa ja sitä, miten toteutus olisi kullakin järkevintä.

Keskityn työssäni siis jaksollisesti toistuviin kuvioihin. Erotan niistä jaksottomasti toistuvat kuviot, vaikka käsittelen niitäkin nopeasti teoriaosuudessa. Tämä rajaus siksi, että itseäni kiinnosti tasaisen kuvion muodostaminen vain yhdellä toistettavalla elementillä. En myöskään käsittele ornamentteja, vaan keskityn enemmän geometrisiin ja pelkistettyihin kuvioihin.

Käyn työn alussa läpi hieman kuvioiden nimityksiä, sillä minulla oli ajoittain hankaluuksia niiden kanssa. Näitä hankaluuksia oli muun muassa lähdekirjallisuutta etsiessä, tai suomentaessa englanninkielisestä lähteestä.

Tämän jälkeen työ jatkuu teoriaosuudella, jossa käsittelen kuvioiden rakennetta. Miten ne muodostuvat, millaisia elementtejä niissä on, miten niitä voi muokata ja miten niitä toistetaan? Käyn lisäksi hieman läpi geometristä taustaa, mutta enemmän jaksottomas-

ti toistuvien kuvioiden osalta, sillä niissä se ilmenee paremmin. Käsittelen myös hieman harmoniaa ja rytmiä tässä yhteydessä.

Teorian jälkeen siirryn kertomaan omista tavoitteistani ja luonnosteluvaiheesta. Tavoitteena minulla on luoda mahdollisimman monikäyttöinen kuvio. Haluan, että se toistuu tasaisesti ja että sen yksittäistä toistuvaa osaa on mahdollisimman vaikea erottaa. Pyrin ottamaan huomioon kuvion tulevan käyttötarkoituksen: onko se kirjan kuvitukseen, nettikäyttöön vai esimerkiksi jonkin pakkauksen koristepinnaksi.

Käyn luonnostelun yhteydessä läpi eri kokeiluja, mitä teen. Pohdin myös eri inspiraatiolähteitäni. Tuon esille luonnoksia, joista lopulta päädyn käyttämään yhtä lopullisena kuvionani.

Seuraavaksi toteutan kuvion käyttäen Adobe Illustratoria, joka on minulle entuudestaan tuttu ohjelma. En ole ennen tehnyt toistuvaa kuviota, mutta tiedän siitä huolimatta melko tarkkaan, mitä aion tehdä ja miten. Tämä vaihe on tästä syystä melko suoraviivainen.

Toteutan saman kuvion Processing-ohjelmointikielellä, joka on täysin erilainen media. Käytän työssä Processingia ensimmäistä kertaa, joten siihen liittyy enemmän harjoittelua ja kokeilua. Kuvaan työnkulun Processingilla, kuten teen Illustratorillakin.

Lopuksi pohdin hieman näiden kahden median eroavaisuuksia. Vertaan niiden hyviä ja huonoja puolia, ja mietin, miten niistä voisi saada parhaan hyödyn. Voisiko niitä mahdollisesti yhdistää?

Säännöllisesti toistuvista kuvioista ei ole laajalti teoksia ilman, että ne painottuvat johonkin tiettyyn alueeseen, esimerkiksi matematiikkaan. Tällöin ne vaativat yleensä syvempää perehtymistä aiheeseen, jotta niitä ymmärtää. Koin siksi aiheelliseksi yhdistää toistuvien kuvioiden perusteorian ja toteuttamisen samaan työhön.

Vektoreiden käyttö on yleistä graafisessa suunnittelussa niiden siisteyden ja helpon muokattavuuden vuoksi, joten se valikoitui automaattisesti ensimmäiseksi mediaksi, jolla tutkia toistuvan kuvion toteutusta. Processing-ohjelmointikieli taas ei ole kovin

yleisessä käytössä. Vaikka Processing on tarkoitettu visuaaliseen ohjelmointiin, ei visuaalisen materiaalin tuottaminen sillä ole kovin yleistä. Ohjelmointi mielletään vaikeaksi

ja etäiseksi, mutta se on nykyään graafisella alalla kysytympi ja arvostetumpi taito. Muiden visuaalisen materiaalin tuottamisen taitojen yleistyessä voi edes perus ohjelmointitaidoilla erottua joukosta. Koska Processing on visuaalinen ohjelmointikieli, voisi se toimia siltana tavallisen suunnittelijan ja ohjelmoinnin välillä, osoittaen, ettei ohjelmointi ole mahdotonta oppia.

2 Yleisesti toistuvista kuvioista

Tässä kappaleessa käsittelen kuvioita yleisesti, sekä teoriaa niiden taustalla. Käyn läpi eri tapoja muodostaa kuvion elementti, ja eri toistamistapoja. Tutkin myös hieman niiden matemaattisuutta, symmetriaa ja harmoniaa.

2.1 Nimityksiä

Aloittaessani työn ja perehtyessäni aiheeseen huomasin pian, että toistuville kuvioille ei ole itsestään selvää, universaalia nimitystä. Havaitsin esimerkiksi, että suomenkieliselle nimelle kuvio on paljon muitakin merkityksiä, yleisin niistä kaavio. Tai vastaavasti, että englanninkielinen sana pattern voi tarkoittaa mitä vain visuaalisesta psykologiseen tai taloudelliseen. Tiedonhaun kannalta tämä oli valitettavaa, sillä aiheeseen liittyvän aineiston erottaminen vääristä hakutuloksista oli välillä hankalaa ja vei ylimääräistä aikaa.

Toistuvia kuvioita voidaan kutsua kuoseiksi, jolloin ne yleensä viittaavat tekstiileihin (suomisanakirja.fi, 2012[A]). Kuosi tai pattern tarkoittaa kuitenkin tekstiilialalla helposti myös kaavaa (sanakirja.org, 2012). Ornamentti-sanalla päästään myös lähelle toistuvan kuvion merkitystä, mutta ornamentti viittaa tietynlaiseen tai tietynhenkiseen kuvioon. Ornamentti mielletään yleensä hyvin koristeelliseksi ja ne ovat monesti vain vertikaalisti tai horisontaalisti toistuvia kuvioita (suomisanakirja.fi, 2012[B]).

Etuliitteillä voidaan yrittää tarkentaa halutun kuvion merkitystä, mutta nekin voivat olla harhaanjohtavia. Toistuva kuvio tai jatkuva kuvio voi edelleen koskea mitä vain muutakin tieteenalaa, vaikka niistä käy ilmi kuvion jatkumo. Painokuvio, printtikuvio, graafinen kuvio tai visuaalinen kuvio rajaavat kaikki kuvion luonnetta visuaaliselle puolelle, mutta niistä ei taas käy ilmi kuvion haluttu toistuvuus.

Olen päätynyt itse käyttämään nimityksiä jaksollisesti toistuva kuvio, säännöllisesti toistuva kuvio tai lyhyemmin vain toistuva kuvio. Toistuva kuvio ei välttämättä tarkoita symmetrisesti toistuvaa, mitä itse haen. Kuvio voi esimerkiksi olla äärettömästi suurentuva kuvio, jolloin se toistuu mutta ei symmetrisesti. Myös saumattomasti toistuva kuvio (eng. seamless pattern) käy tarkoitukseeni, sillä se viittaa visuaaliseen kuvioon, ja myös jaksollisesti toistuvaan, sillä siinä on kyse kahdesta kuvion elementistä, jotka jat-

kuvat toisistaan ilman näkyvää saumakohtaa. Englanniksi sana tiling eli laatoitus on myös omaan tarkoitukseeni toimiva, mutta se on helposti sidottu geometriaan, joten se ei päde yleisnimitykseksi.

2.2 Tapoja toistaa kuvioita jaksollisesti

Lähdemateriaalia selatessani listasin, että toistuvia kuvioita voidaan jaotella erilaisiin kategorioihin, kuten geometrisiin tai matemaattisiin kuvioihin, ornamentteihin ja tietyille aikakausille tai kulttuureille ominaisiin kuvioihin.

Näiden kaikkien tyyppien tutkiminen on kuitenkin mahdotonta aiheen rajauksen kannalta. Mielestäni yhtenäisin tapa tutkia kuvioiden sommittelua on avata rakenne, josta ne muodostuvat, sillä kolme hyvinkin erilaista kuviota voivat muodostua kaikki samasta rakenteesta tai pohjasta, jota on vain toistettu eri rytmissä tai määrissä (Bunce & Phillips 1993, 6). Siksi olenkin valinnut sen lähestymistavakseni aiheen teoriaan.

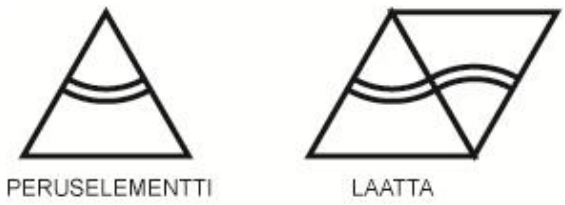
Toistuville kuvioille on olemassa useita erilaisia tapoja, joilla niitä voidaan sommitella. Esimerkiksi kuutiotoistossa (block repeat) kuvio toistuu suorassa linjassa niin pystykuin vaakatasossa. Se on toistotavoista kaikista yksinkertaisin ja sitä voi varioida esimerkiksi etäisyyksiä muuttamalla. Pudotustoistossa (drop repeat) kuvio putoaa puolet korkeudestaan seuraavalla pystyrivillä. Tätäkin voi muunnella pudottamalla kuviota esimerkiksi vain neljänneksen korkeudestaan. Tiiliskivitoisto (brick repeat) on pudotustoistoa vastaava, mutta kuvio siirtyy seuraavalla vaakarivillä puolet leveydestään sivuun. Epäsäännöllisessä toistossa (irregular repeat) kuvio ei noudata yhtä selkeää sääntöä, vaan siinä voi olla useampi variaatio pudotus- tai tiiliskivitoistosta, tai se voi muuttaa suuntaa välillä. (Bunce & Phillips 1993, 26–131.) Näitä esimerkkisommittelupohjia esiintyy eri aikakausilta olevissa kuvioissa. Vaikka kuvioiden pinta ja ulkonäkö muuttuvat kulttuurista ja ajasta riippuen, pysyvät niiden perusrakenteet samana.

Kuvioon vaikuttaa toistotavan lisäksi myös sen kokonaissommitelma. Kuvio voi olla pintasommitelma, jolloin se jatkuu kaikkiin suuntiin. Kuvio voi myös olla nauhasommitelma, jolloin se toistuu jatkuvasti kahteen suuntaan joko vertikaalisti tai horisontaalisti. Ornamentit ovat yleisin ryhmä nauhamaisia sommitelmia. Kun kuvio ei ole sellaisenaan

jatkuva, vaan on rajoiltaan määritelty, sitä kutsutaan täytesommitelmaksi. (Pusa 1979, 136-138.)

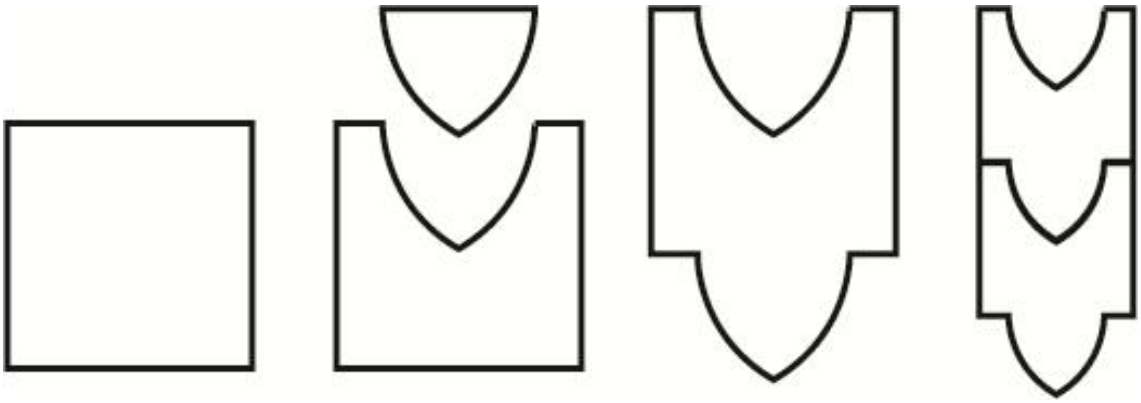
2.3 Peruselementti ja laatta

Kuvion toistamisessa symmetrisesti vaikuttaa toistettava elementti suuresti sen lopputulokseen. Jaksollisessa toistossa toistettavan elementin, jota kutsun laataksi, täytyy olla muoto, joka toistuu aukottomasti ilman, että sitä tarvitsee kääntää. Toisin sanoen laatta ei voi olla esimerkiksi kolmio, ympyrä tai viisikulmio, sillä ne eivät toistu aukottomasti. Muotoja, jotka voivat sellaisenaan toimia laattoina, on suorakulmiot, vinoneliöt ja kuusikulmiot. Sen sijaan peruselementti voi olla kolmio tai epäsäännöllinen nelikulmio, josta muokkaamalla tehdään laatta, jota sitten toistetaan kuviossa (kuvio 1). Näitä tapoja valmistaa peruselementistä laatta on neljä, ja ne on esitelty kappaleissa 2.3.1, 2.3.2, 2.3.3 ja 2.3.4. (Beyer 1999, 17-19.) Symmetrisellä peruselementillä tulee helposti melko yksitoikkoista jälkeä, mutta sen ollessa epäsymmetrinen sillä on paljon enemmän vaihtoehtoja. Silloin siihen voi vaikuttaa kääntämällä elementtiä, tekemällä siitä peilikuvan, yhdistämällä sen seuraavaan tietystä kohdasta jne. Näin hyvinkin yksinkertaisella peruselementillä voi saada mielenkiintoisen ja vaikeastikin hahmotettavan kuvion.



Kuvio 1. Peruselementti voi olla minkä muotoinen tahansa, mutta laatan täytyy olla toistettavissa sellaisenaan.

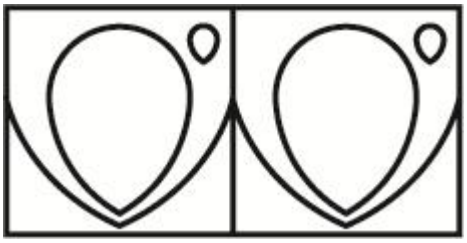
Vaikka laatan muoto on rajattu, voidaan sitä hallitusti muuttaa lomittain toistuvaa kuviota tehdessä. Tämä tapahtuu siten, että valitun laatan (kuviossa 2 neliön) yhdestä tai useammasta reunasta irrotetaan halutun muotoinen pala ja se siirretään tai käännetään toiseen kuvion reunaan. Kuvioita toistettaessa reuna, johon siirrettty pala on lisätty sopii saumattomasti reunaan, josta pala on otettu pois. (Beyer 1999, 4.)



Kuvio 2. Neliölaatasta poistetaan pala, ja se liitetään toiseen reunaan. Nyt laatat ovat lomittain toistuvia.

2.3.1 Translaatio

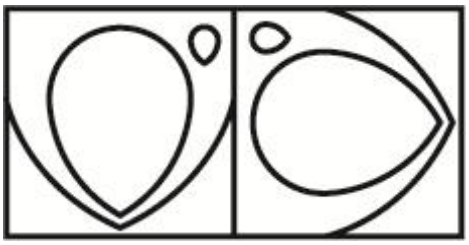
Ensimmäinen tapa muodostaa peruskuviosta laatta on translaatio (Kuvio 3). Translaatiossa peruskuvio on kuitenkin sama kuin laatta, ja sitä voi liikuttaa pysty- tai vaakasuunnassa, jolloin kyseessä on joko horisontaali tai vertikaali translaatio. (Beyer 1999, 20.)



Kuvio 3. Horisontaali translaatio

2.3.2 Rotaatio

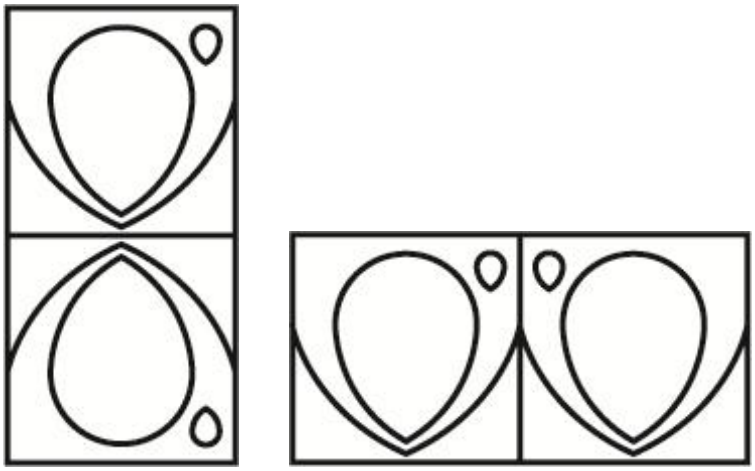
Toinen tapa on rotaatio, jossa peruskuviota käännetään akselinsa ympäri (Kuvio 4). Sitä voi kääntää yhden tai useamman kerran ja eri kulmissa riippuen sen muodosta, kunhan lopputuloksena syntyy laatta. Esimerkiksi tasasivuista kolmiota tulisi kääntää kuusi kertaa, jolloin siitä muodostuu kuusikulmio. (Beyer 1999, 20.)



Kuvio 4. 90 asteen rotaatio

2.3.3 Peilaus

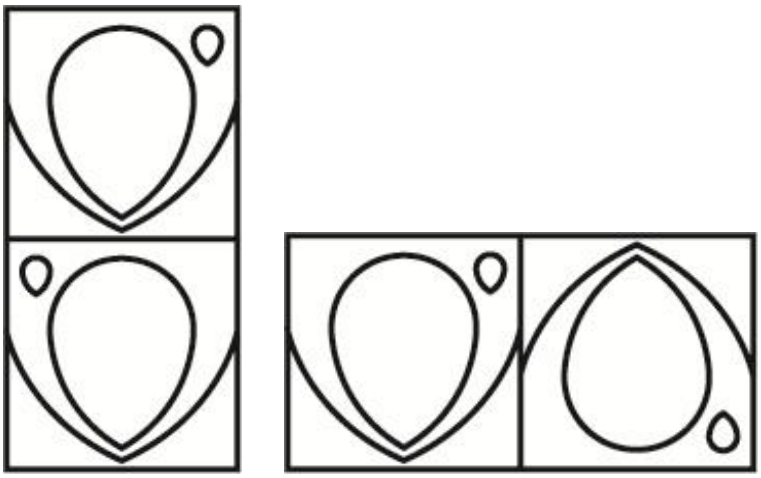
Kolmas tapa on tehdä peilikuva, jossa peruselementistä syntyy käännetty kopio (Kuvio 5). Peilauksen voi tehdä sekä horisontaalisti että vertikaalisti. (Beyer 1999, 20-21.)



Kuvio 5. Horisontaali peilaus sekä vertikaali peilaus

2.3.4 Liuku

Viimeinen tapa tehdä peruskuviosta laatta on liu'uttamalla (Kuvio 6). Horisontaalisti liu'uttamalla peruselementti siirretään ensin horisontaalisti ja sitten se peilataan vertikaalisti. Vastaavasti vertikaalisti liu'uttamalla peruselementti ensin siirretään vertikaalisti ja sitten peilataan horisontaalisti. (Beyer 1999, 20-21.)



Kuvio 6. Horisontaalisti liu'utettu sekä vertikaalisti liu'utettu elementti

2.4 Tason säännöllinen jakaminen

M.C. Escher määritti 17 sääntöä tason säännölliseen jakamiseen, eli 17 symmetriaryhmää. Nämä ryhmät perustuvat edellä mainittuihin tapoihin valmistaa peruskuviosta laatta. Näille on olemassa standardilyhenteet, jotka ovat esimerkiksi muotoa p4 tai p3m1. P4 tarkoittaa, että peruselementti käännetään neljä kertaa 90 astetta valitun kulman ympäri. P3m1 taas tarkoittaa, että peruskuvio ensin peilataan ja sitten käännetään kolme kertaa. (Beyer 1999, 26-27.) Merkintöjen ymmärtäminen on aluksi hankalaa ja vaatii tottuneempaa säännöllisesti toistuvien kuvioiden tulkintaa. Alla on listattu 17 sääntöä lyhyillä selityksillä. Kaikki alla mainitut tehdään peruselementille, joka on kolmion tai neliön muotoinen

1	P1	Translaatio (kuviolle ei tapahdu mitään)
2	P2	Käännetään 180-astetta
3	P4	Käännetään 90-astetta neljä kertaa
4	PG	Liu'utus joko horisontaalisti tai vertikaalisti
5	PGG	Liu'utetaan kaksi kertaa, sekä horisontaalisti että vertikaalisti
6	PM	Peilataan
7	CM	Peilataan ja syntynyttä peilikuvaa siirretään, joko pudotus- tai tiiliskivitoistolla

8	PMG	Sama kuin ylempi, lisäksi vielä liu'utetaan
9	PMM	Peilataan ensin horisontaalisti sitten vertikaalisti
10	CMM	Sama kuin ylempi ja siirretään pudotus- tai tiiliskivitoistolla
11	P4G	Käännetään neljä kertaa ja sitten liu'utetaan
12	P4M	Peilaus, jonka lisäksi käännetään neljä kertaa
13	P3	Käännetään kolme kertaa
14	P6	Käännetään kuusi kertaa
15	P3M1	Peilataan, jonka jälkeen käännetään kolme kertaa
16	P31M	Käännetään kolme kertaa ja peilataan
17	P6M	Peilataan, jonka jälkeen käännetään kuusi kertaa

(Beyer 1999, 26-27.)

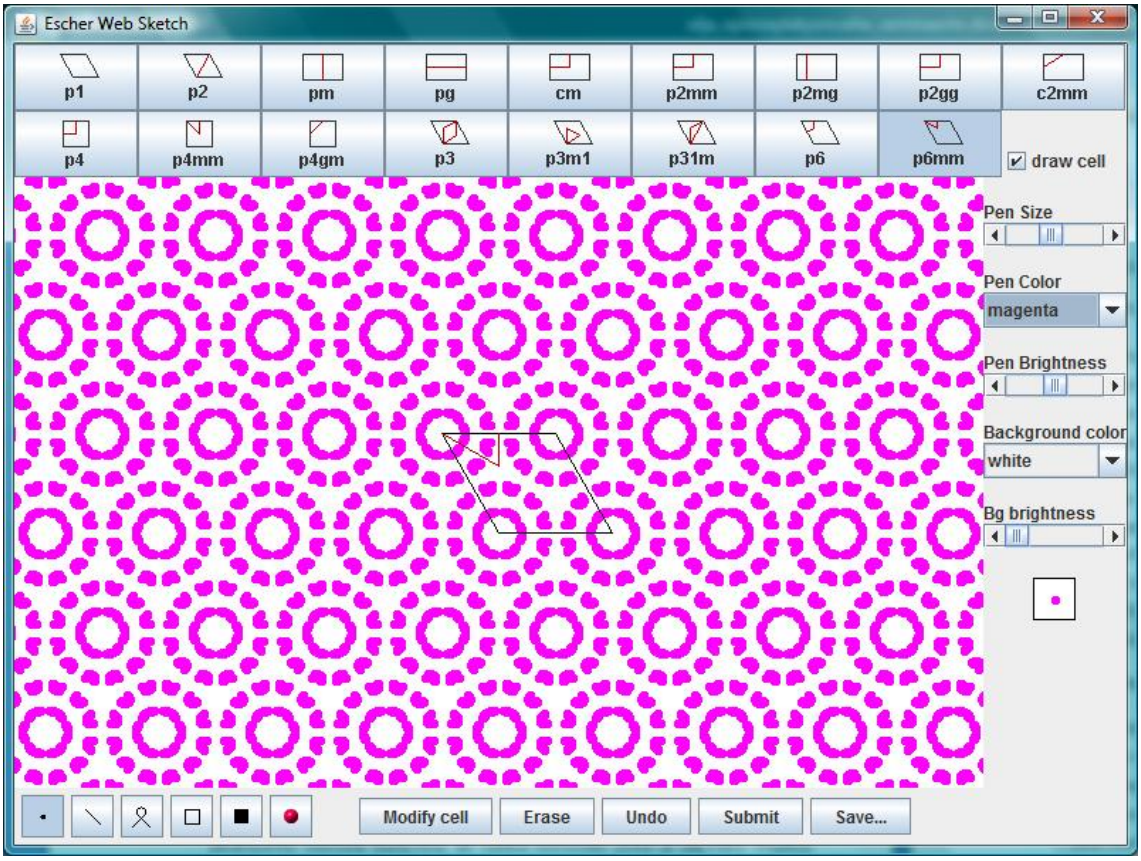
2.4.1 17 symmetriaryhmän keksijä M.C. Escher

M.C. Escher on 1898 Hollannissa syntynyt kuvataiteilija. Hän opiskeli grafiikan tekniikoita Haarlemin arkkitehtuuri- ja taidekoulussa. (Ernst 1978, 7.) Escher on tunnettu matemaattisista ja mahdottomia asioita kuvaavista töistään, mutta hänellä ei ollut matemaattista koulutusta. Escherille oli siksi haastavaa luoda systeemi tason säännöllistä jakamista varten. Systeemi tunnetaan nykyään hänen nimellään. Escherin tutkimus tason jakamisesta on herättänyt mielenkiintoa kristallografien ja matemaatikkojen keskuudessa. Escher hyödynsi tason jakamista sykli-töissään ja metamorfoosi-töissään, joissa matemaattiset kuviot muuttuvat eläimiksi, taloiksi tai muiksi tunnistettaviksi muodoiksi. (Ernst 1978, 20.) Escherin kiinnostus tason säännölliseen jakamiseen syntyi Alhambrassa, jossa hän jäljensi ja tutki maurilaisia ornamentteja (Ernst 1978, 35-36).

2.4.2 Escher web sketch

Escher Web Sketch on yksinkertainen Java-pohjainen sovellus, johon on ohjelmoitu Escherin kehittämät 17 tasoryhmää. Sovelluksessa voi valita tasoryhmän ja piirtää siihen vapaalla kädellä tai lisäten valmiita muotoja. Sovellus toistaa piirrettyä kuviota samanaikaisesti koko pinta-alalle. Sovellus ei ole kovin visuaalinen eikä muutenkaan sovellu tarkempaan työskentelyyn, mutta se on tehokas ja nopea tapa hahmottaa tasoryhmiä. Kuviossa 7 on sovellus ja siihen piirretty nopea kuvio kahdella lyhyellä viivalla.

Sovelluksella on helppo suunnitella lukuisia luonnoksia ja saada virhekokeilujenkin kautta lisää ideoita eri mahdollisuuksista. Sivulla on myös pari vinkkiä, miten saada helposti tietty lopputulos.



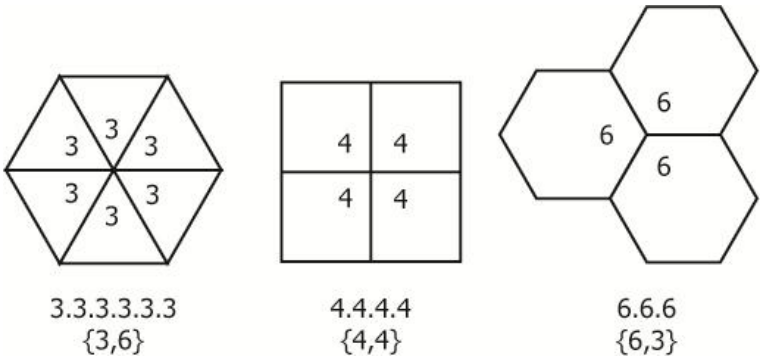
Kuvio 7. Escher Web Sketch sovelluksen näkymä

2.5 Geometria kuvioiden takana

Käsittelen tässä työssä kuvioita visuaalisesta näkökulmasta ja olen rajannut niiden matemaattisen puolen vähemmälle. Päädyin tähän, sillä se on niin laaja aihe tutkittavaksi, että se vaatisi oman työnsä. Toistuvien kuvioiden matemaattisuutta on tutkittu paljon, etenkin jaksottomasti toistuvien kuvioiden, joten en yritäkään käsitellä sitä kaikkea tässä. Esittelen kuitenkin Penrosen laatat, sillä niihin ei voi olla törmäämättä toistuvia kuvioita tutkittaessa ja ne ovat mielestäni huomionarvoisia sellaisista puhuttaessa. Lisäksi esittelen aiheeseen läheisesti liittyvän kultaisen kolmion.

Käyn myös läpi hieman perusgeometriaa. Aiemmassa kappaleessa (2.3 Peruselementti ja laatta) mainitsin jo hieman jaksollisesti toistuvien kuvioiden taustalla olevia geometrisia muotoja. Säännöllisesti toistuvissa kuvioissa esiintyviä muotoja ovat ainakin tasasivuinen kolmio, vakiomuotoiset suorakulmiot, vakiomuotoiset suunnikkaat ja säännölliset kuusikulmiot (Kivelä 2000, monikulmiot 3/5). Viisi- ja kymmenkulmiot ja vinoneliö ts. rombi ovat myös osana toistuvia kuvioita, mutta ne liittyvät jaksottomaan toistettavuuteen (lisää kappaleessa 2.5.1 Penrosen laatat ja 2.5.2 Kultainen kolmio.)

Matematiikassa on määritetty kolme säännöllistä kuviota (regular tesselation). Nämä ovat tasasivuinen kolmio, neliö ja säännöllinen kuusikulmio. Ne ovat valikoituneet siitä, että ne toistuvat saumattomasti ilman erillisiä välikappaleita. Kun nämä muodot asetetaan yhden pisteen ympärille, niiden yhteen osuvien kulmien summa on 360-astetta. Näille muodoille on yhteistä siis se, että niiden kaikki kulmat ovat yhtäsuuret. Tästä syystä esimerkiksi vinoneliötä ei lasketa joukkoon, vaikka sekin toistuu saumattomasti. Muiden tasakylkisten monikulmioiden muodot eivät puolestaan täytä 360-astetta tasalukuna, eli kokonaisina muotoina. Nämä kolme kaikki kriteerit täyttävää muotoa on nimetty niiden kulmien mukaan. Kuusikulmiossa on kuusikulmaa, jotka toistuvat kolme kertaa täyttääkseen 360-astetta, joten se on nimetty 6.6.6. tai {6,3}. Neliössä on neljä kulmaa ja se toistuu neljä kertaa pisteen ympäri täyttääkseen 360-astetta, joten sen nimitys on 4.4.4.4 tai {4,4}. Kolmiolla vastaavasti 3.3.3.3.3.3 tai {3,6}. Nämä on esitetty kuviossa 8. (Totally Tesselated 2012.)

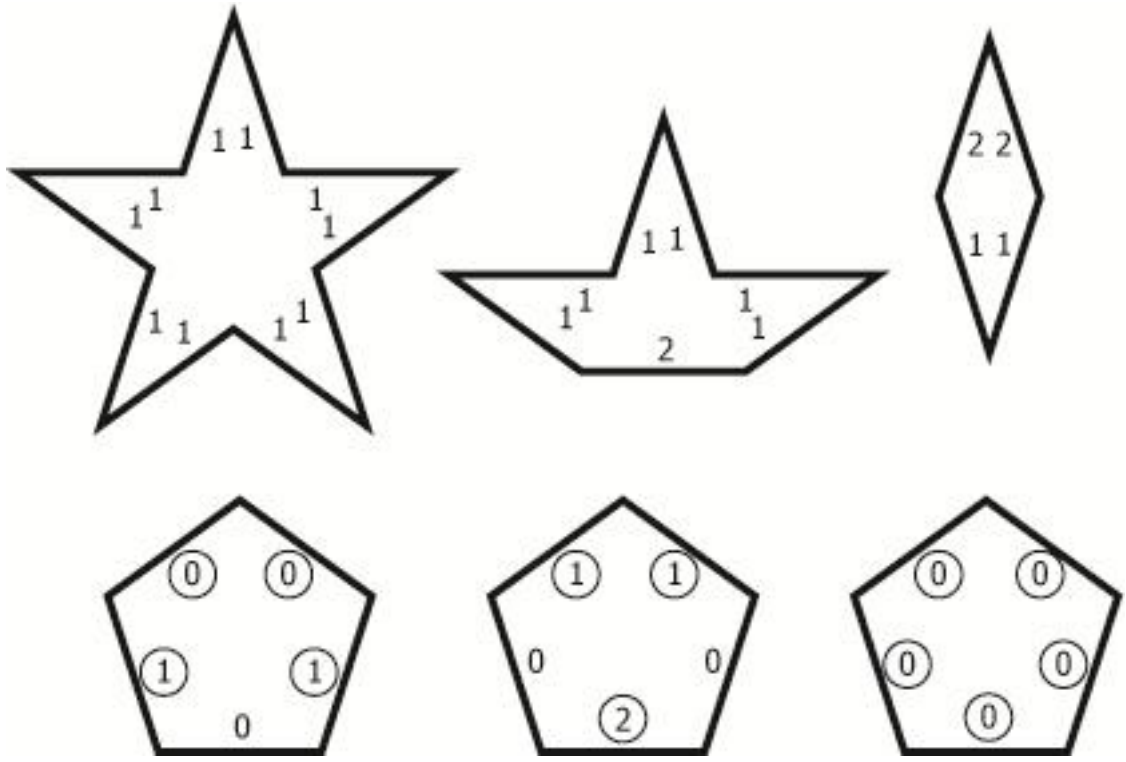


Kuvio 8. Matematiikassa määritetyt kolme säännöllistä kuviota.

2.5.1 Penrosen laatat

Matemaatikko Roger Penrose tutki geometrisiä kuvioita ja muodosti niistä ryhmiä, joita voi toistaa määrättyjen sääntöjen mukaan jaksottomasti. Nämä kuvioryhmät nimettiin hänen mukaansa ”penrose tiles” eli penrosen laatat. Laattoja on kolme eri joukkoa, jotka kaikki koostuvat monikulmioista. (Grünbaum & Shepherd 1987, 531.)

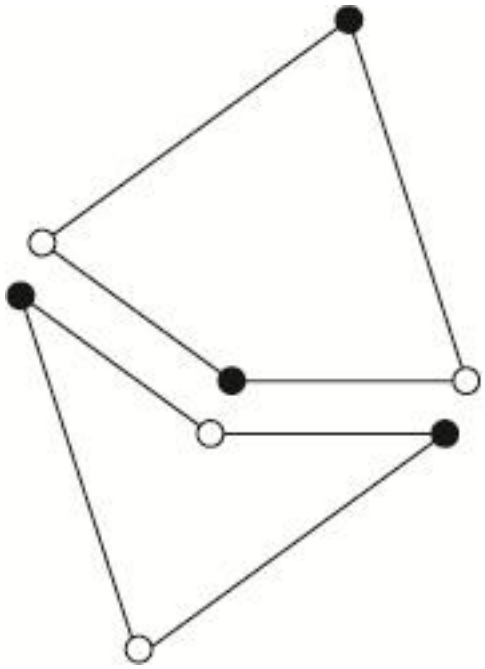
Ensimmäinen näistä joukoista on P1 ja se koostuu kuudesta osasta. Ne ovat kolme tasasivuista viisikulmiota, tähti, vene (ts. 3/5 tähdestä tai osittaisesta tähdestä, josta puuttuu kaksi sakaraa) ja timantti eli rombi. Näillä kaikilla on ns. nimetyt sivut ja tiettyjen sivujen täytyy aina olla määrättyä paria vasten (esitetty kuviossa 9). (Grünbaum & Shepherd 1987, 531.)



Kuvio 9. P1 joukon kuviot. Sivun, joka on merkitty 1 täytyy tulla vasten sivua, jossa 1 on ympyröity. Vastaavasti 0 vasten sivua, jossa 0 on ympyröity ja sama numerolle 2.

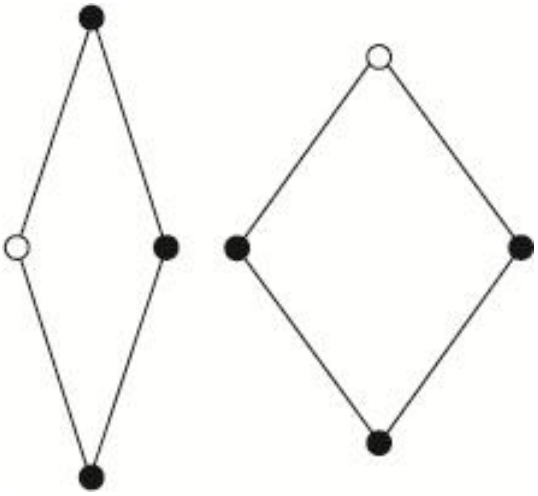
Toiseen joukkoon, P2, kuuluu leija- ja nuolenpääkuvio (kites and darts). Leija on nelikulmio, jonka kulmat ovat 72, 72, 72 ja 144 astetta, ja nuolenpää on nelikulmio, jonka kulmat ovat 36, 72, 36 ja 216 astetta. Nämä kaksi muotoa koostuvat kahdesta kultaisesta kolmiosta (selitetty kappaleessa 2.5.2). Leija- ja nuolenpääkulmiot muodostavat yhdessä tasasivuisen nelikulmion. (Kivelä 2000, monikulmiot 5/5) Näihin pätevän tois-

tumissäännön mukaan vain tietyt kulmat voivat muodostua vertexin ympärille. Kulmia on kahta eri ja ne on merkitty kuviossa 10 eri värisin ympyröin. (Grünbaum & Shepherd 1987, 536-538.)



Kuvio 10. P2 joukon leija ja nuolenpää. Näitä toistaessa tulee mustalla merkittyjen kulmien aina osua saman verteksin ympärille, ja valkosten vastaavasti aina samaan.

Kolmas joukko eli P3 (kuviossa 11) koostuu kahdesta rombista, jotka ovat keskenään eri levyiset. Nämä rombit ovat samankaltaisia P2 joukon kuvioiden kanssa ja ne voidaankin muodostaa näistä kuvioista. Täten myös P3 joukon kuvioissa esiintyy kultainen kolmio. (Frettlöh 2006, Penrose Rhomb.)



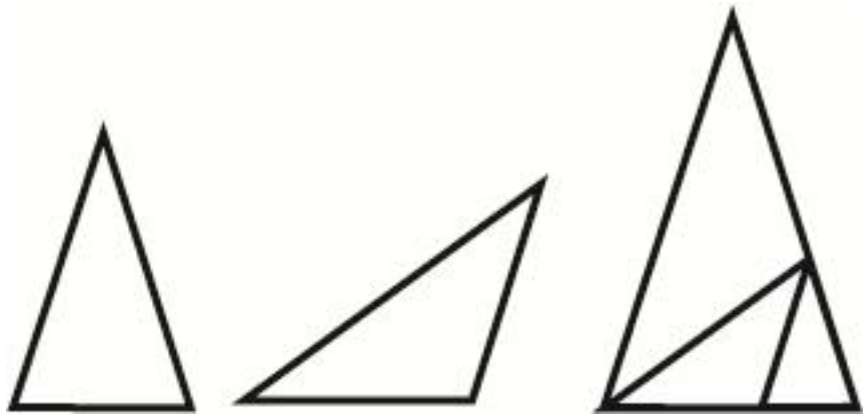
Kuvio 11. Viimeisen joukon P3 rombit. Niihin pätee sama kuin joukkoon P2, eli mustalla merkityt kulmat saman verteksin ympärille ja valkoiset keskenään saman.

Penrosen laatoitusta näkyy arkkitehtuurissa maailmalla, mutta lähemmin myös Helsingissä. Sitä käytettiin kun Keskuskatu muutettiin Stockmannin kohdalta kävelykaduksi, ja se sai uuden laatoituksen sen yhteydessä. (Järvinen 2009, 68.)

2.5.2 Kultainen kolmio

Kultainen kolmio on tasakylkinen kolmio, jonka toisen kyljen suhde lyhyeeseen sivuun on kultainen leikkaus. Kultainen kolmio on esimerkiksi viisisakaraisen tähden yksi sakara. Sen saa muodostettua, kun säännöllisen kymmenkulmion kaksi vierekkäistä kulmaa yhdistää kulmion keskipisteeseen. (Weisstein 2012, Golden Triangle.)

Robinsonin kolmiot ovat kaksi erimuotoista kolmiota (kuvio 12). Molemmat ovat tasakylkisiä, mutta niillä on erisuuruiset kulmat. Robinsonin kolmioista tekee erikoisen se, että ne toistavat itseään. Toinen kolmioista, teräväkulmainen kolmio, on kultainen kolmio. Toinen niistä on kultainen gnomoni, joka on tylppäkulmaisempi. Kultainen kolmio muodostuu, kun sen toisen tasakyljen yhdistää kultaisen gnomonin toiseen tasakylkeen. Vastaavasti kultainen gnomoni muodostuu, kun sen toisen tasakyljen yhdistää kultaisen kolmion lyhyeeseen sivuun. (Weisstein 2012, Golden Triangle.)



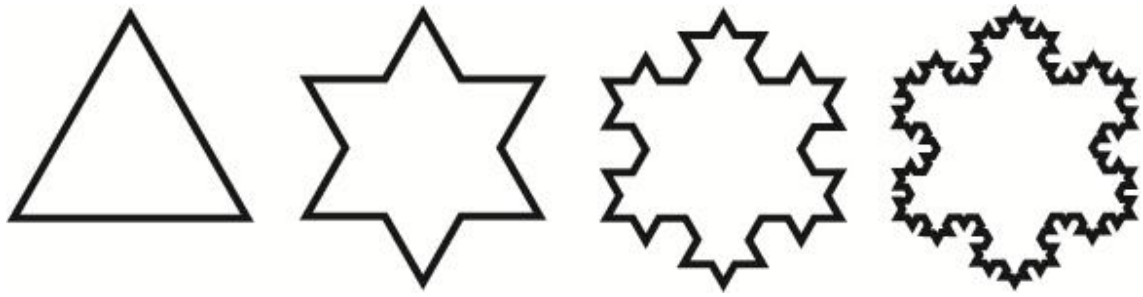
Kuvio 12. Kultaisen kolmion ja kultaisen gnomonin jakoa kutsutaan Robinsonin kolmioiksi.

Penrosen laatat perustuvat kaikki samojen muotojen ympärille. Yhdistelemällä yhden joukkion kuvioita, voi saada aikaiseksi jonkun toisen joukkion kuvion. Kaikkiin näihin muotoihin pätee siis myös samanlainen toistuvuus. Kaikkia näitä joukkoja on mahdollista toistaa jaksollisesti, mutta koska niitä tutkiessa on haettu jaksottomuutta, niille on asetettu yhteensopivuussääntöjä. Niitä ei saa yhdistää tavalla, jolla voi muodostaa jaksollisesti toistuvan elementin. (Grünbaum & Shepherd 1987, 531-547.)

Penrosen laatat eivät kuitenkaan muodosta täysin satunnaista kuviota, vaan niissä on tietynlainen toistuvuus. Niiden erikoisuus on se, että haluttaessa ne toistuvat loputtomasti yhdensuuntaisesti, eli keskeltä ulospäin. Penrosen tutkimat kuviot ovat itseään toistavia (kuten Robinsonin kolmiotkin). Esimerkiksi viisikulmion voi jakaa pienempiin osiin, jolloin se muodostuu kuudesta viisikulmiosta. Tällaisesta sisäkkäisestä toistumisesta johtuen ulospäin rakentuva kuvio toistaa itseään äärettömästi – aina vain isompana ja isompana. (Hwang 2012, Penrose Tilings.)

2.6 Ääretön yhdensuuntainen toistuminen

Penrosen laatat voivat rakenteensa vuoksi muodostaa äärettömästi jaksottomasti toistuvan kuvion. Näiden lisäksi on muita äärettömiä toistuvuuksia, kuten fraktaalit. Esimerkiksi Kochin-käyrän tai lumihiutaleen (kuvio 13) alku on yksinkertainen tasasivuinen kolmio, jonka sivuista tulee uudet tasasivuiset kolmiot, jotka ovat kooltaan kolmasosan alkuperäisestä. Näiden kolmen uuden kolmion sivuista tulee taas alkuperäisestä kolmasosan olevat uudet kolmiot, joiden sivuista tulee edelleen uudet. (Helmberg 2007, 4-5.)



Kuvio 13. Kochin käyrä/lumihiutale

Toinen esimerkki on puu, jonka runko haarautuu kahteen oksaan. Nämä oksat puolestaa haarautuvat jälleen kahteen oksaan , jotka haarautuvat edelleen kahteen oksaan. Oksat ovat aina kaksi kolmasosaa edellisen mitasta, jolloin myös tämä voi jatkua ja pienentyä äärettömästi.

Sen lisäksi, että Escher tutki tason jakamista, hän oli myös kiinnostunut mahdottomista perspektiiveistä. Tämän kiinnostuksen aikana hän tutustui Penrosen kolmioon, joka antoi idean Escherin yhdelle tunnetuimmista töistä, nimeltään Vesiputous. Penrosen

kolmio, eli tribar, on Roger Penrosen tunnetuksi tekemä kolmio. Se on kaksiulotteisella pinnalla ikäänkuin loputon kolmio, mutta olisi kolmiulotteisena mahdoton, sillä se perustuu vääriin liittymiin. (Ernst 1978, 87-88.) Tämän kiinnostuksen myötä Escher päätyi tutkimaan äärettömyyttä kaksiulotteisella tasolla. Tätä tutkiessaan Escher loi yhdet mielenkiintoisimmista töistään, eli Circle Limit -sarjan työt. Niissä hän toistaa tavallista laatoitusta hyperbolisella tasolla. Hyperbolisessa geometriassa eivät päde samat säännöt kuin euklidisessa geometriassa (taso- ja avaruusgeometria). Matemaatikko Poincare esitti kaavan, jossa ääretön taso rajoitettiin ympyrään. Tästä Escher keksi uusia ideoita äärettömyyden kuvaamiselle, eli laatoitus jatkuu äärettömästi ympyrän reunoilla, sillä se pienenee koko ajan lähestyessä reunaa. (Ernst 1978, 107-109.) Escher myös jatkoi tätä teemaa Neliö-limiitti töissään, joissa on sama idea, mutta kuvio jatkuu äärettömästi neliön reunoja kohden.

2.7 Harmonia, symmetria ja rytmi sommittelussa

Kuvion toistuessa tasaisesti ja staattisesti katsojan silmä saa levätä, sillä vaikutelma on rauhallinen ja harmoninen. Toistuvissa kuvioissa harmonia syntyy samankaltaisuuksista niin värien kuin muotojen suhteen. Harmoniselle vaikutukselle on ominaista kuvion sopusuhtaisuus ja se, että yhtäläisyydet ovat merkittävämpiä kuin erilaisuudet. Yksityiskohtien ja kokonaisuuden välillä tulee olla tyylillinen johdonmukaisuus, ettei voi syntyä epäsymmetriaa ja irrationaalisuutta. (Pusa 1979, 117-130.) Harmoniaa voi olla, vaikka muodot olisivat erilaiset, kunhan sitä tasapainottaa koon vaihtelulla tai väreillä. Tasapaino voi olla symmetrinen tai epäsymmetrinen ja syntyä muodoista tai valööristä, kunhan se luo tunteen samanarvoisesta painotuksesta tai huomionarvosta. (Heikerö 2001, 25-27.)

Symmetria on toinen tärkeä tekijä tasaisessa toistuvassa kuviossa. Symmetrialla voi vaikuttaa siihen, kuinka dynaaminen tai staattinen kuvio on. Staattinen kuvio syntyy yleensä perinteisellä symmetrialla ja tasavälisellä ja -muotoisella toistolla. Se voi parhaimmillaan olla hyvin yksinkertaista. Epäsymmetria puolestaan luo elämää ja liikettä. Jos kuvio tai sen toisto on epäsymmetristä, on se heti paljon monipuolisempi ja dynaamisempi. (Pusa 1979, 117-130.) Myös rytmi vaikuttaa kuvion toistuvuudessa. Rytmillisesti kertaamalla kuvioita tai kuvioryhmiä eri välein voi luoda erilaisia toistuvuuksia ja vaikutelmia. (Pusa 1979, 133-138.)

3 Jaksollisen kuvion toteutus Adobe Illustratorilla ja Processing-ohjelmointikielellä

Käyn seuraavaksi läpi vaiheet, jotka liittyvät säännöllisesti toistuvan kuvion toteutukseen. Aloitan tavoitteistani ja kerron sitten luonnostelusta. Sen jälkeen kerron vaihe vaiheelta molemmista valitsemistani toteutustavoista, Adobe Illustratorista sekä Processing-ohjelmointikielestä. Lopuksi hieman vertailen niitä keskenään.

3.1 Tavoitteet ja ideat

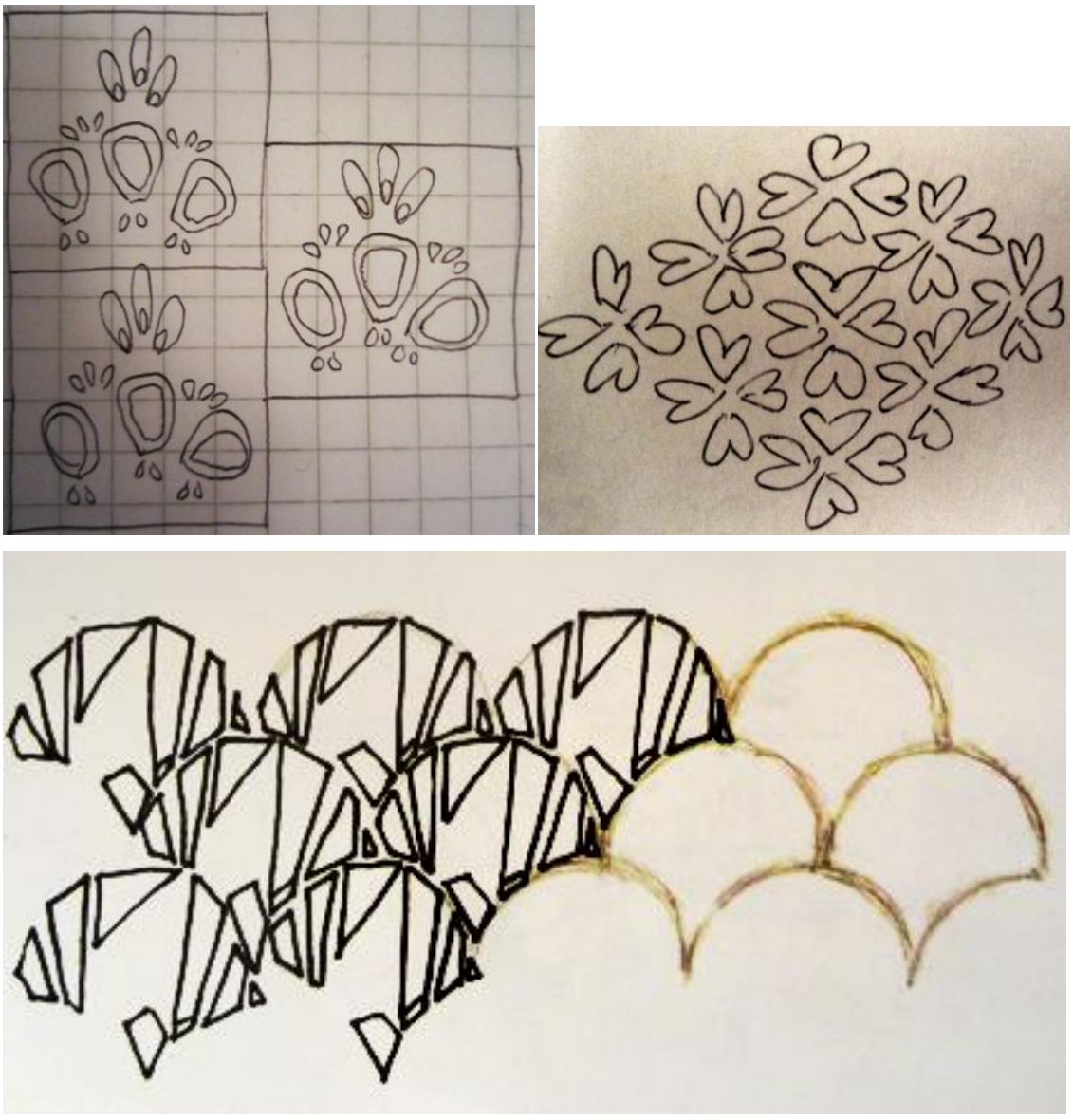
Olin päättänyt etukäteen, että en toista kuviossa jotain yhtä selkeästi erottuvaa elementtiä, vaan yritän saada kuviot sekoittumaan mahdollisimman tasaisesti toisiinsa. Halusin myös että kuvio toistuu jaksollisesti. Halusin kuitenkin jonkinlaisen pääelementin, koska yksi lähtökohtani on luoda mahdollisimman monikäyttöinen kuvio. Jos kuviota käyttää esimerkiksi jossain tietyssä taitossa, voisi kyseisen toistuvan pääelementin nostaa muillakin tavoin siinä esiin, luoden yhtenäisyyttä kokonaisuuteen. Mikäli kuvio olisi nettisivulla, vaikka taustalla, ei se saisi olla liian räikeä muodoiltaan tai väreiltään, ettei se ärsytä katsojaa tai häiritse sivuston sisältöä. Jos se toistuisi pakkauksen pinnalla, pitäisi siinä olla jotain mielenkiintoista, mikä kiinnittäisi huomion pakkaukseen. Kuvion tuleva käyttötarkoitus tulee siis ottaa ajoissa huomioon, jotta kuvio sopii siihen luontevasti.

Minua viehättävät hyvin yksityiskohtaiset kuviot, ja siksi haluaisin omassanikin olevan mahdollisimman paljon pieniä yksityiskohtia. Yksityiskohdat auttavat kuvion toistuvia osia sekoittumaan huomaamattomammin toisiinsa.

Halusin kuvion, joka on mahdollista toteuttaa käyttäen valittua vektorigrafiikkaohjelmaa, eli Adobe Illustratoria, sekä pyrkiä toteuttamaan saman kuvion myös ohjelmoiden Processing-ohjelmointikielellä. Kuvio on mahdollista toteuttaa molemmilla keinoilla, mutta niissä on eri hyödyt sekä haitat. Halusin toteuttaa kuvion molemmilla tavoilla, sillä halusin saada kuvioiden teosta mahdollisimman paljon tietoa ja kokemusta. Näissä kahdessa on täysin erilainen lähestymistapa kuvion toteutukseen, ja näin ne antavat paljon laajemman kuvan kuvion teosta.

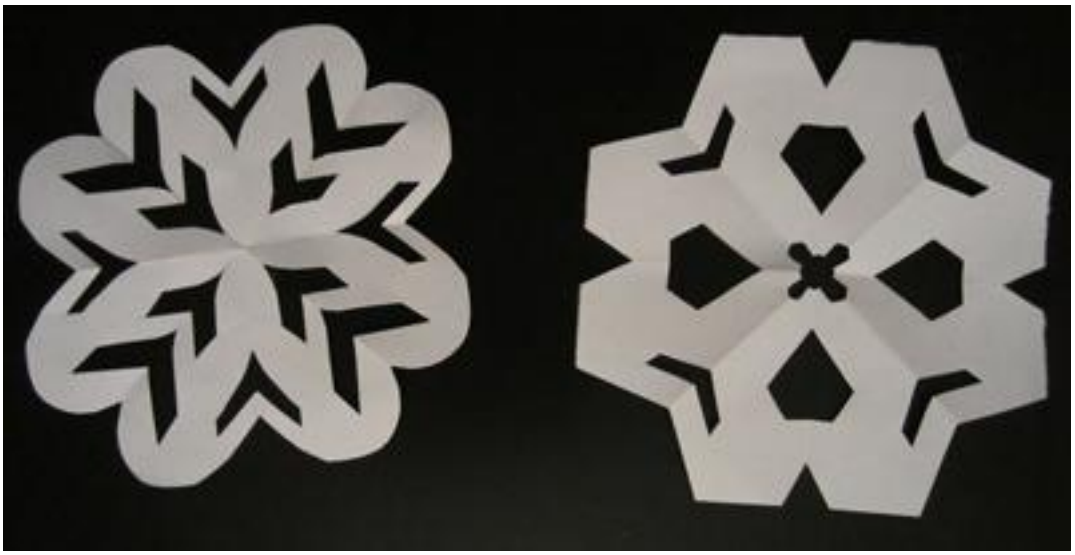
3.2 Työn luonnostelu

Minulla oli mielessäni selkeät tavoitteet, mitä haluan kuviolta, mutta vain hatara idea siitä, millainen se käytännössä voisi olla. Luonnostelin aluksi käsin paperille useita erilaisia kuviomalleja, joista osa näkyy kuviossa 14. Minulla oli suuria vaikeuksia keksiä toimivaa elementtiä, joten luonnostelin käsin, koska se oli kaikista nopein tapa minulle kokeilla ideaa pikaisesti. Välillä skannasin jonkin mahdollisen kuvion ja jäljensin sen Illustratoriin toistoa varten, mutta päädyin toteamaan, ettei se toimikaan.



Kuvio 14. Esimerkkejä käsin hahmotelluista ideoista.

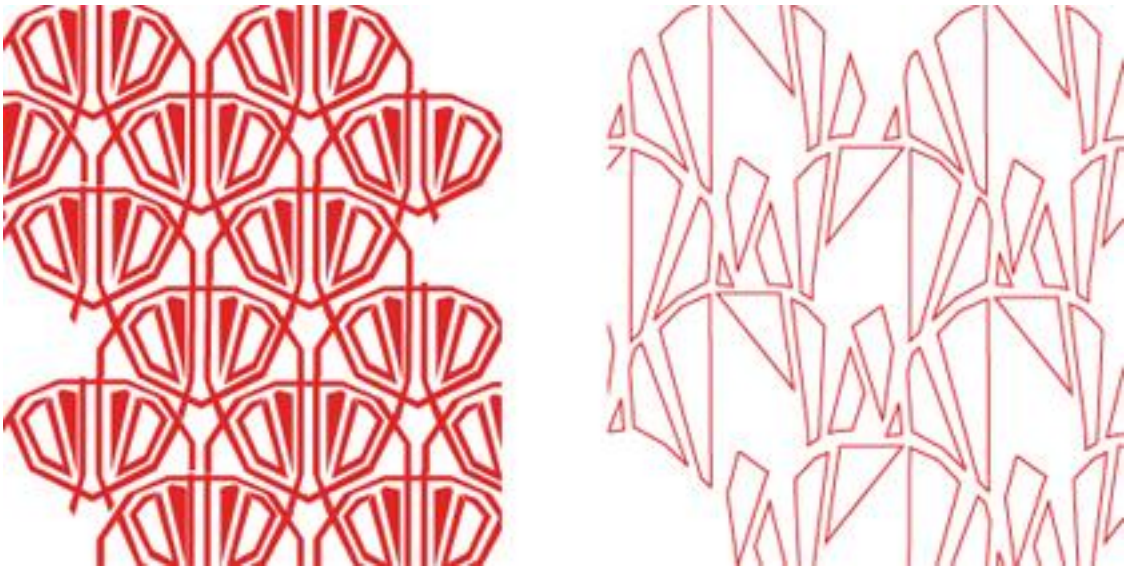
Käydessäni läpi erityyppisiä ratkaisuja mieleeni tuli lapsuudesta tutut paperista leikatut lumihiutaleet. Otin palan paperia ja taitoin sitä muutaman kerran puoliksi ja jälleen puoliksi, ja leikkaisin sitten reunoilta paloja pois. Kun avasin paperin, oli siinä symmetrinen kuvio, jota toistamalla olisi saanut tasapainoisen kuviopinnan, kuten kuviossa 15 näkyy.



Kuvio 15. Paperista taiteltuja ja leikattuja symmetrisiä kuvioita, ts. lumihiutaleita.

Minulla oli etukäteen mielessä tietynlaisia muotoja, joita olisin halunnut käyttää kuviossa. Ne olivat kaikki pyöreän muotoisia elementtejä, esimerkiksi pisara tai sydän. Pyöreä muoto tuo kuvioon leikkisyyttä ja jotain kiinnostavaa, mutta se on myös hyvin silmiin pistävä. Elementtiä on huomattavasti vaikeampi saada sopimaan tasapuolisesti osaksi kuviota, sillä se on selkeästi rajattu muoto, ja siksi se vaikuttaa helposti silmään yksittäiseltä elementiltä, vaikka sen ympärillä olisi paljon muutakin.

Päädyin viemään luonnoksista eteenpäin suorakulmiosta muokattua, limittäin toistuvaa muotoa. Muoto on viuhkamainen, melko yksinkertainen, ja siksi siihen päädyinkin. Halusin tässä vaiheessa selkeän peruskuvion, jolla lähteä liikkeelle. Kokeilin tästä muutamaa variaatioita, kun yritin hakea sopivaa sisältöä kuviolle. Osan toteutin Illustratorilla, ja osan Processing-ohjelmointikielellä. Illustratorilla tehdyistä kokeiluista yhdessä käänsin joka toisen rivin peilikuvaksi yrittäessäni tuoda kuvioon jotain pientä lisää. Toisessa hain jotain, mikä yhdistäisi viuhkaelementit toisiinsa (Kuvio 16).



Kuvio 16. Kaksi erityyppistä mutta samaan pohjaan Illustratorilla tehtyä kuviota.

Processing-kokeiluissa (kuvio 17) toisessa loin vain peruspohjan, mutta se oli niin silmää ärsyttävä aaltoilullaan, että luovutin sen heti alkuun. Toisesta kokeilusta, jossa viuhkat toistuvat diagonaalisti tulikin melkein lopullinen kuvioni. Koin sen kuitenkin lopulta niin tylsäksi, että luovuin siitäkin.

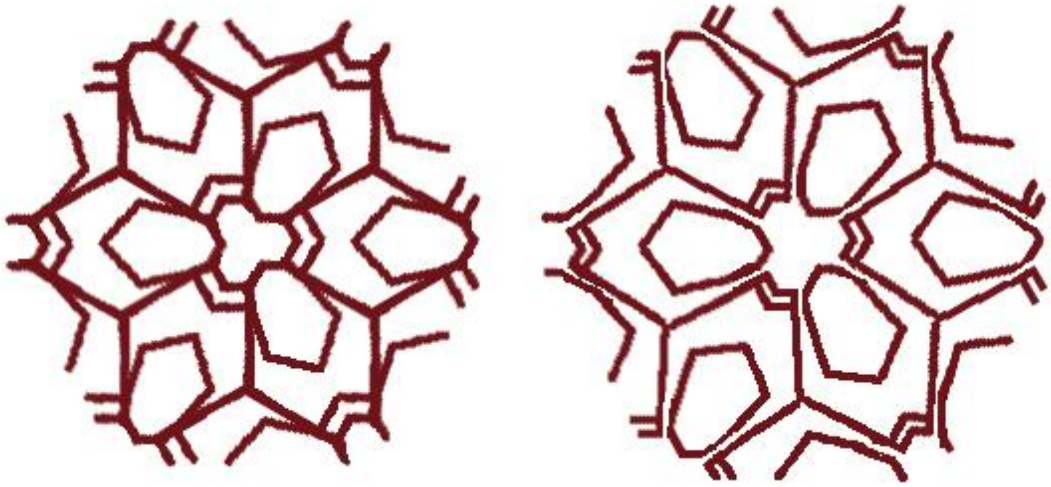
Kuvio 17. Processing kokeiluja, kaikki viuhkamuodosta inspiroituneita.

Tässä vaiheessa epätoivo alkoi nostaa päätään. En päässyt eroon itselleni asettamista rajoista ja tavoitteista, minkä takia pyöritin samoja muotoja uudestaan ja uudestaan. Päätin siksi aloittaa vielä kerran aivan alusta. Palasin miettimään, mitkä asiat minua viehättävät ja inspiroivat ja mitä en halua kuviooni. Päätin ainakin luopua täysin viuhkamaisesta pohjasta. Minua viehättivät suuresti kaleidoskooppimaiset kuviot, joissa

yksinkertaisesti toistetaan paria viivaa peilaamalla ja kopioimalla niin, että syntyy kokonaisuus, joka on mielenkiintoinen toistettaessa.

Selailtuani erityylisiä kuvioita eri aikakausilta tiesin, että minua viehättävät myös islamilaiset geometriset kuviot, erityisesti esimerkiksi moguli-kauden aikaiset. Ne ovat ankaran geometrisia ja vaikka niissä olisi monimutkainen kuvio, vähäisten värien ja yksinkertaisten muotojen ansiosta ne näyttivät tasaisilta ja harmonisilta. Tämä johtuu siitä, että niissä on kekseliäästi toistettu samoja muotoja monin eri tavoin, jolloin ne ovat visuaalisesti yhtenäisiä ja kuitenkin mielenkiintoisia katsoa. Nämä muodot ovat yleisimmin monikulmio ja tähti, jossa on viisi, kuusi, kahdeksan, kymmenen tai kaksitoista sakaraa. Myös oktagoni ja ns. puolikastähti toistuvat monissa kuvioissa. Arkkitehtuurissa kuvioihin on tuotu värikontrastia yhdistämällä valkoista marmoria punaiseen hiekkakiveen. Lisää värivariaatioita on saatu myös keltaisella hiekkakivellä ja mustalla kivellä. (Michell 2007, 68-70.) Mielestäni näillä vähäisillä väreillä on saatu kauniita pintoja aikaan, kun ne tulevat esille yksinkertaisissa geometrisissa muodoissa.

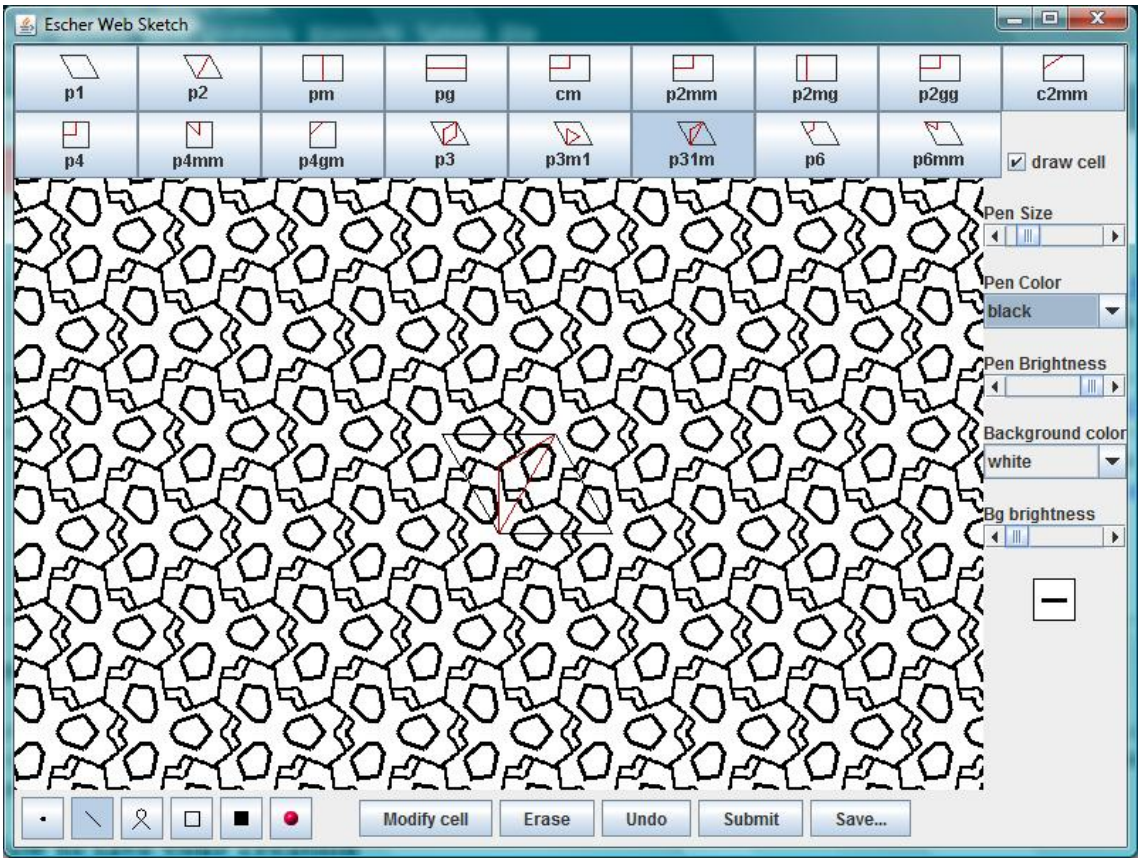
Halusin ottaa näistä inspiraatiota, mutta en kuitenkaan halunnut tehdä selvästi tämän tyyppistä kuviota, sillä tuntui, etten saisi niistä tarpeeksi omaperäisiä. Lopulta sain idean, kun luin uudestaan läpi symmetriasääntöjä (esitelty kappaleessa 2.4). Niistä sääntö p31m kiinnitti huomioni. Se miellytti melkein kaleidoskooppimaisuudellaan, mutta myös sillä, että siitä saisi helposti toisiinsa limittäin yhteensopivan kuvion. Kuvioni sopii p31m-symmetriasääntöön, koska siinä puolikkaan timantinmuotoinen kolmio kopioidaan ja käännetään 120 astetta kaksi kertaa, jolloin se muodostaa tasasivuisen kolmion. Tämä kolmio vielä peilataan muodostamaan timantinmuotoinen kappale. (Beyer 1999, 77.) Omassa kuviossani tätä timanttia kopioidaan ja käännetään vielä kaksi kertaa muodostamaan kuusikulmio. Lisäsin tähän pohjaan poimimani inspiraation islamilaistyylisten kuvioiden muodoista ja väreistä, ja viimein löysin kuvion, jossa on tasapainossa tavoittelemani asiat (kuvio 18).



Kuvio 18. Kaksi variaatiota kuviosta johon lopulta päädyin. Niissä ei ole vielä värejä, mutta muoto on selvillä.

Kuvioni yksittäinen elementti on siis kuusikulmio, joka lepää yhdellä sivulla. Tämä toistuu pudotustoistolla (puhuttu kappaleessa 2.2), sillä seuraavan pystyrivin kuvio on puolet alempana kuin edellinen. Pystyrivit menevät hieman limittäin, jotta toistettaessa kuvion kaikki sivut koskevat toisiaan.

Tässä välissä kokeilin vielä luonnostella Escher websketchillä (esitelty kappaleessa 2.4.2). Olin hieman yrittänyt käyttää sitä luonnostellessa, sillä ajattelin, että sillä tavoin saisi tehokkaasti kokeiltua erilaisia ideoita. Sovellus ei kuitenkaan auttanut minua, sillä päädyin vain kokeilemaan erilaisia summittaisia muotoja, jotka eivät oikein johtaneet kunnolliseen lopputulokseen. Sovellus kuitenkin auttaa hahmottamaan Escherin määrittämiä sääntöjä, ja sillä saattoi päätyä yllättäviinkin lopputuloksiin. Kokeilin websketchiä nyt uudelleen, kun minulla oli tietty kuvio jo mielessä. Sain käytettyä sovellusta erilaisesta näkökulmasta nyt, kun tiesin, mitä teen. Tietämällä suunnilleen, mitä haen sain ohjelmasta ehkä enemmän irti, mutta kunnon toteutukseen se ei riittänyt, vaan oli enemmän testaamista varten. Kuviosta toistui silti luonnokseni mukainen kuten kuviossa 19 näkyy.



Kuvio 19. Lopullista kuvio ideaa kokeiltuna Escher Web Sketchillä.

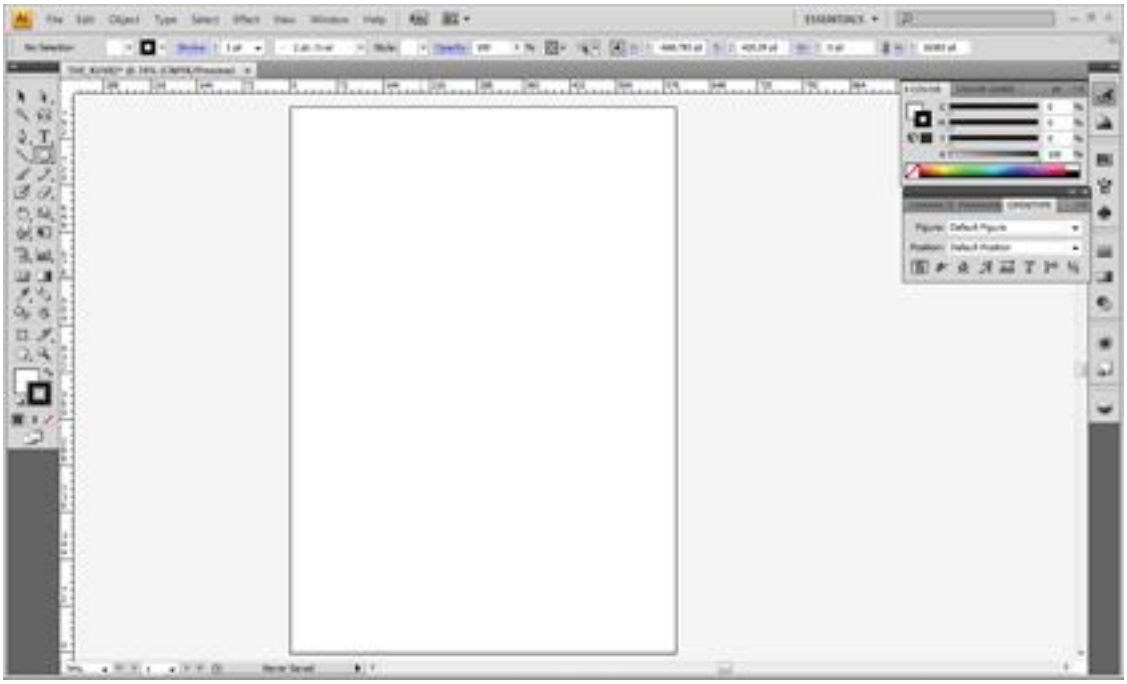
3.3 Mikä on Adobe Illustrator?

Illustrator on Adobe Systemsin kehittämä vektorigrafiikkaohjelma. Se kuuluu nykyään osana Adobe Creative Suite – ohjelmistopakettiin, lyhennettynä CS, ja siitä on tullut kaikenkaikkiaan 15 versiota. Itse käytän työssäni CS4-versiota, joka on 14. versio alusta alkaen laskettuna. (Korkeila 2007, 5.)

Ensimmäisen version Adobe kehitti omaan käyttöönsä ja sitä käytettiin PostScript-tiedostojen muokkaamiseen. Tänäkin päivänä Illustratoria voi käyttää eri PostScript-sivunkuvauskieleen pohjautuvien tiedostojen, mm. PDF-tiedostojen, avaamiseen ja muokkaamiseen. Alunperin Illustrator kehitettiin Mac-käyttöjärjestelmälle, mutta hyvin pian siitä tehtiin myös Windows-käyttöjärjestelmälle sopiva. Omassa työssäni käytän Illustratoria Windows-käyttöjärjestelmällä, mutta se ei vaikuta työn kulkuun, sillä ohjelmat ovat käyttöjärjestelmillä näppäinkomentoja lukuunottamatta täysin samat. (Korkeila 2007, 5.)

Illustratorilla on paljon ominaisuuksia ja sen käyttö voi aluksi tuntua vaikealta. Kuitenkin se mahdollistaa juuri vektorien ansiosta hyvin monipuolisen ja joustavan grafiikan luomisen ja muokkaamisen. Vektorigrafiikassa luodaan ankkuripisteiden ja niiden välille muodostuvien vektoripolkujen avulla objekteja. Objektien ulkoasuun voi vaikuttaa muokkaamalla polun ja sen sisään jäävän pinnan eri ominaisuuksia, kuten väriä tai paksuutta. Vektorigrafiikkaa voi skaalata vapaasti ilman, että se vaikuttaa grafiikan laatuun, sillä vektorigrafiikkaa ei ole sidottu resoluutioon. (Korkeila 2007, 16.)

Illustratorin päänäkymä näkyy kuviossa 20. Keskellä on piirustusalusta ja sen ympärillä valikkoja. Vasemmalla on yleisimmin päätyökalupalkki ja oikealla muut lisätyökalut omissa ikkunoissaan. Ylhäällä on valikko, josta nämä oikean puolen työkalut löytyvät, mikäli niille ominaiset ikkunat eivät ole auki. Tämä on yleisin järjestys, mutta työkalupalkki ja ikkunat ovat siirreltävissä. Omassa työssäni käytän työkalupalkkia sekä ylhäällä olevia valikoita.



Kuvio 20. Illustrator-näkymä. Vasemmalla työkalupalkki, ylhäällä valikot ja oikealla lisää ominaisuuksia.

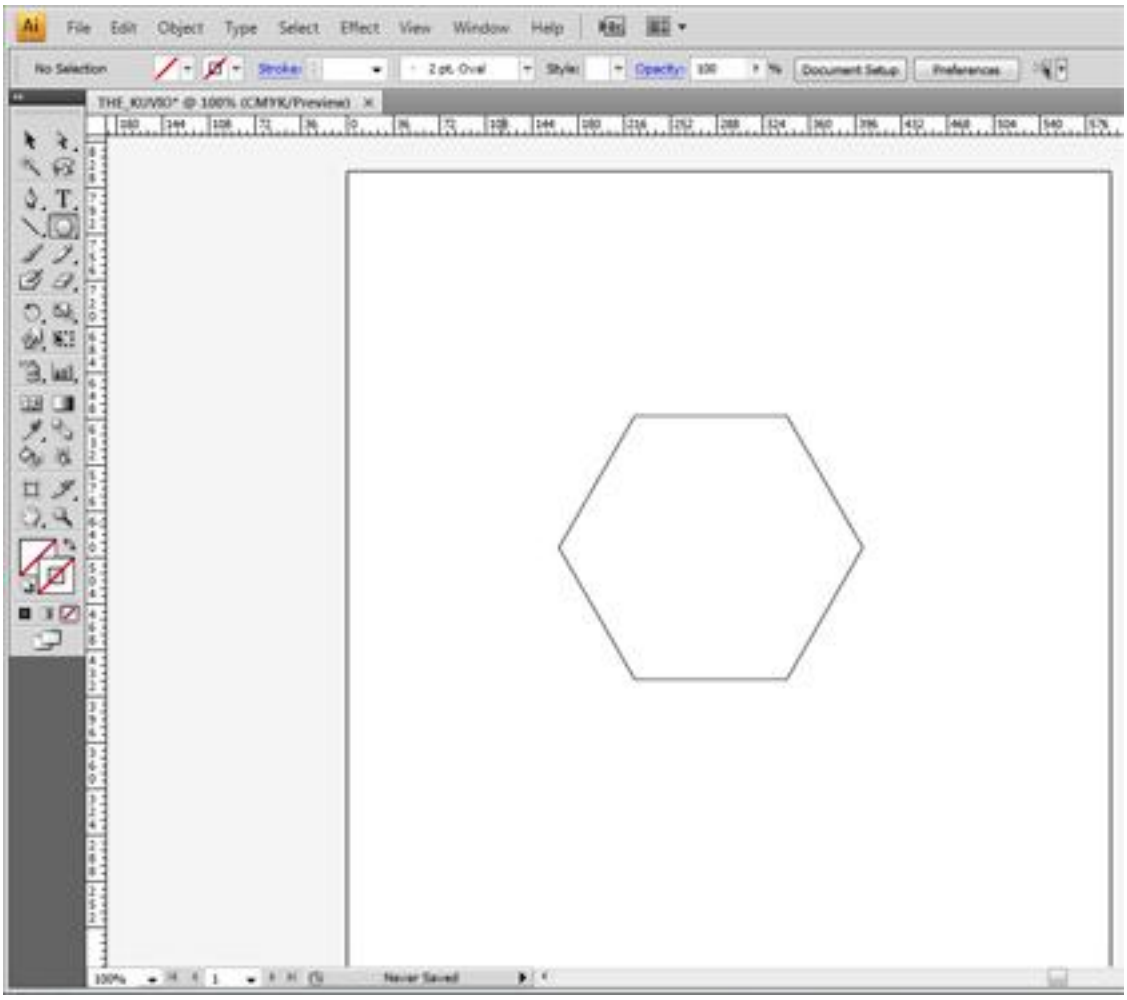
3.3.1 Toteutus Adobe Illustratorilla

Päätin tehdä ensimmäisen toteutuksen Adoben Illustrator -vektorigrafiikkaohjelmalla, sillä se tuntui sopivimmalta tarkoitukseeni. Olen käyttänyt Illustratoria useamman vuoden ajan, joten se on minulle entuudestaan tuttu, ja tiesin siksi, mitä teen.

Illustratorissa käytetään vektoreita. Kuvioni on viivapainotteinen, joten vektorit sopivat hallittavuudeltaan siihen hyvin. Lisäksi ohjelman ominaisuuksien perusteella tiesin, että voisin tehdä mittatarkkaa työtä, mikä on tärkeää moninkertaisesti toistuvan kuvion yhteydessä.

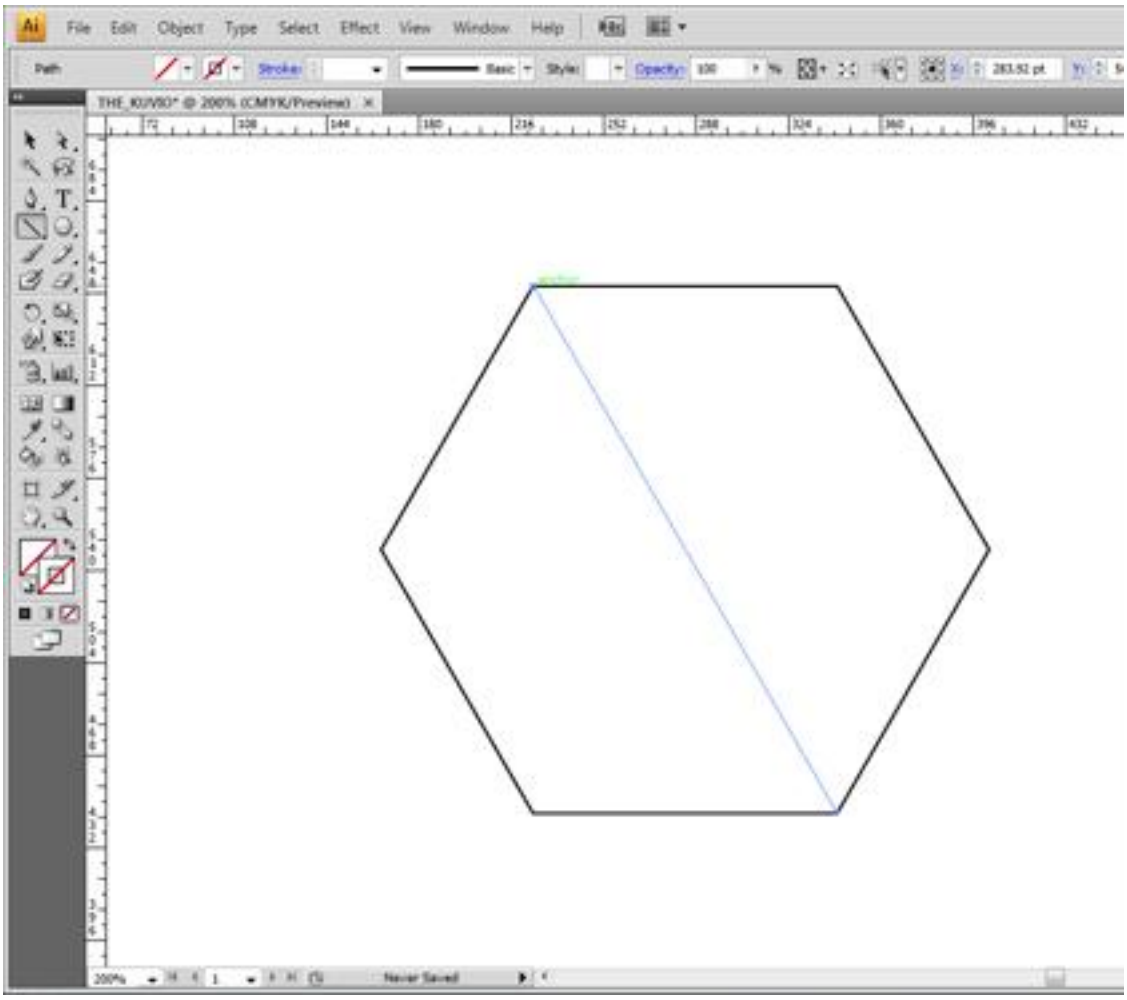
3.3.2 Työvaiheet

Aloitan työn avaamalla tyhjän dokumentin. Piirrän tähän monikulmiotyökalua (polygon tool) käyttäen kuusikulmion klikkaamalla tyhjää pohjaa. Tämä avaa ikkunan johon voi määritellä, kuinka monta kulmaa ja kuinka suuren säteen kulmiolle haluaa. Kuvio syntyy automaattisesti lepäämään suorassa yhdellä kyljellä, kuten kuviossa 21.



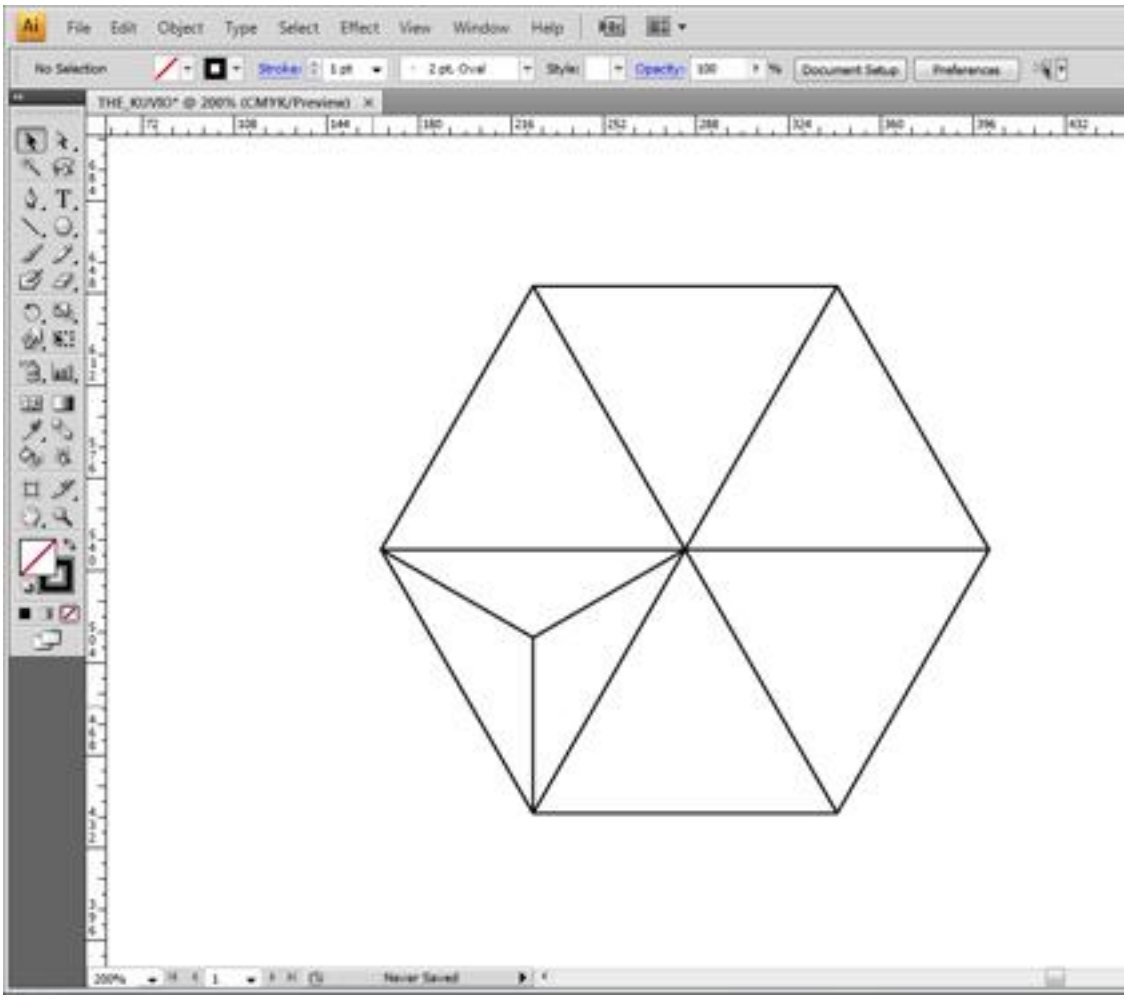
Kuvio 21. Monikulmiotyökalulla piirretty kuusikulmio lepää automaattisesti kyljellään.

Seuraavaksi jaan kuvion kuuteen osaan piirtämällä viivatyökalulla (line segment tool) kuvion yhdestä kulmasta vastakkaiseen kulmaan viivan. Viivan saa tarkasti kulmasta kulmaan, kun varmistaa, että aloittaa ja lopettaa sen jo valmiina olevista ankkuripisteistä (anchor), kuten kuviossa 22. Tämän kanssa pitää koko ajan olla tarkka, tai muuten kuvio alkaa vääristyä. Sitä ei välttämättä huomaa yhden elementin kohdalla, mutta kun se toistuu monta kertaa, pienimmätkin virheet tulevat esille. Kun kuusikulmio on jaettu kuuteen osaan, teen vektoreista ryhmän (group). Tämä helpottaa niiden hallinnointia myöhemmin.



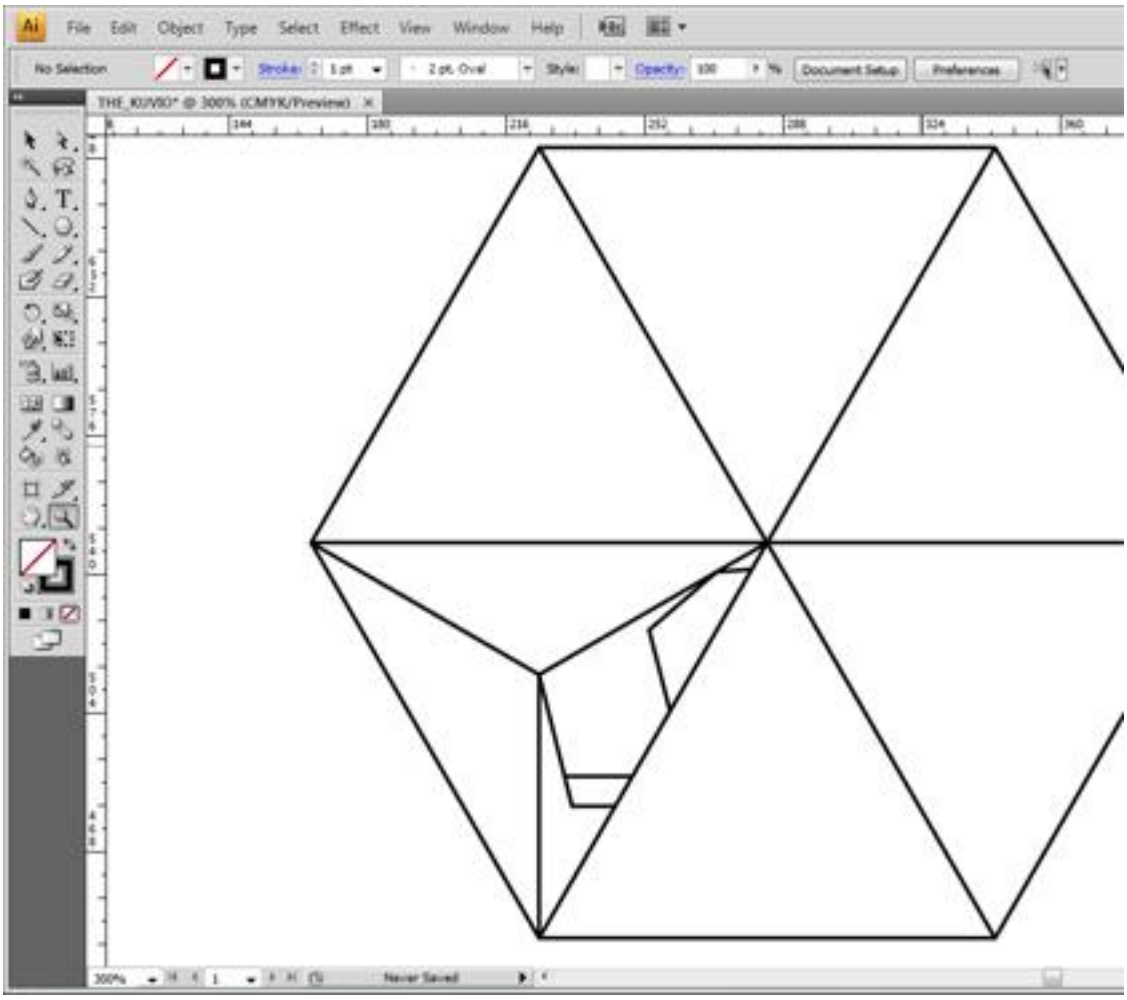
Kuvio 22. Jakaessa kuusikulmion kuuteen osaan, täytyy viivan alkaa ja loppua olemassa olevaan ankkuripisteeseen.

Jaan seuraavaksi vielä yhden syntyneistä kuudesta kolmiosta kolmeen osaan, joihin itse toistuva osa piirretään. Koska kuvioni toistuva osa peilautuu pitkän sivunsa mukaisesti, jaan kolmion kulmista kolmeen osaan. Aloitan viivan piirtämisen kulman ankkuripisteestä ja vedän viivan suoraan vastapäätä olevan sivun keskelle. Kun toistan saman kaikille kulmille, viivat risteävät keskellä. Lyhennän sitten kaikki viivat tähän keskipisteeseen, jotta jäljelle jää kolme kolmiota, ja teen niistä ryhmän (kuvio 23). Kaikki tähän mennessä piirretyt viivat ovat apuviivoja. Koska ne on ryhmitetty, ne on helppo poistaa sitten, kun kuvio on piirretty.



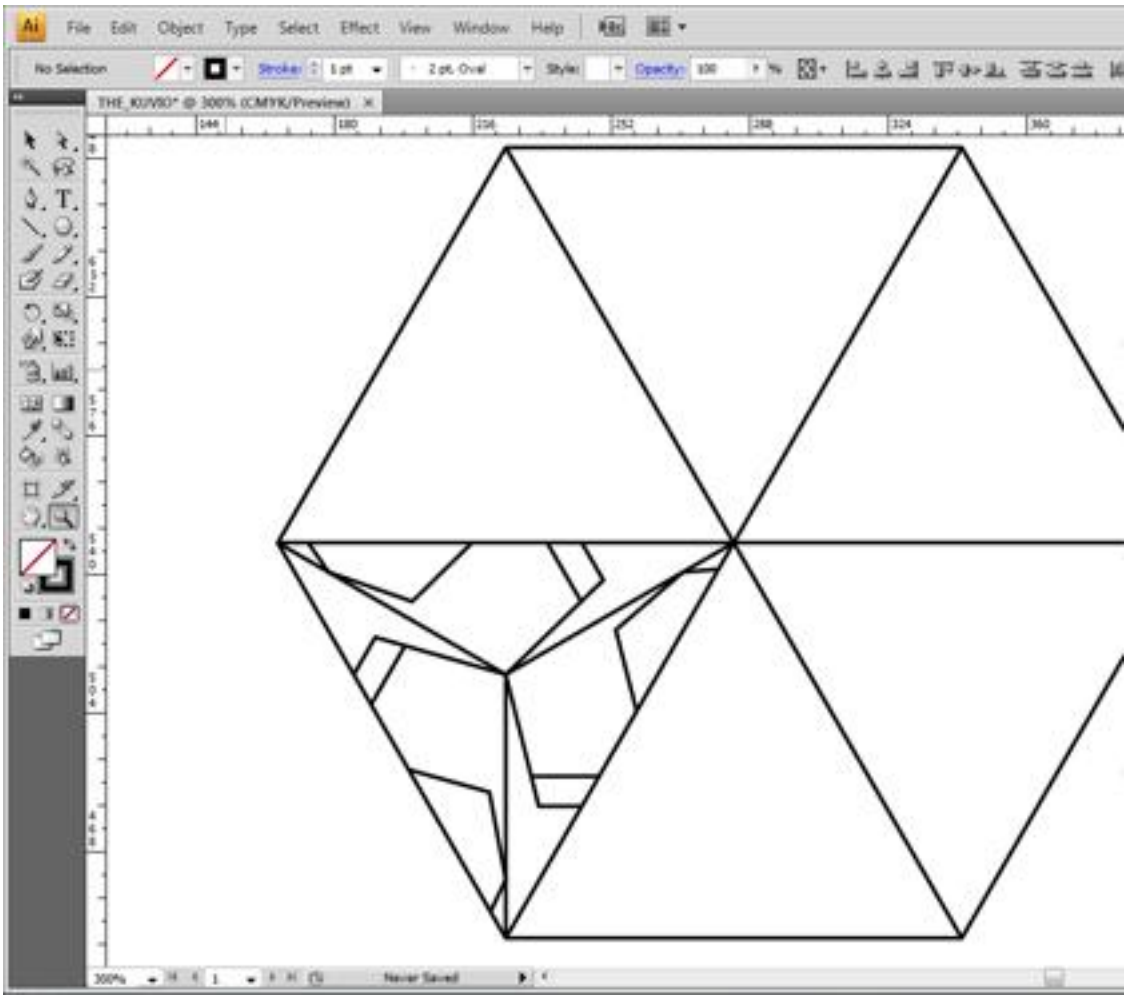
Kuvio 23. Syntynyt kolmio jaetaan edelleen kolmeen symmetriseen osaan.

Nyt on valmiina se pieni alue, jota kääntämällä ja peilaamalla syntyy kokonainen elementti, jota sitten toistetaan luomaan koko kuviopinta. Haluan varmistaa, että kaikki piirtämäni viivat ovat varmasti symmetrisiä, joten piirrän siksi vain yhden kolmion verran kuviota (kuvio 24), ja käännän sen tietyllä asteluvulla sitten muihin osiin.



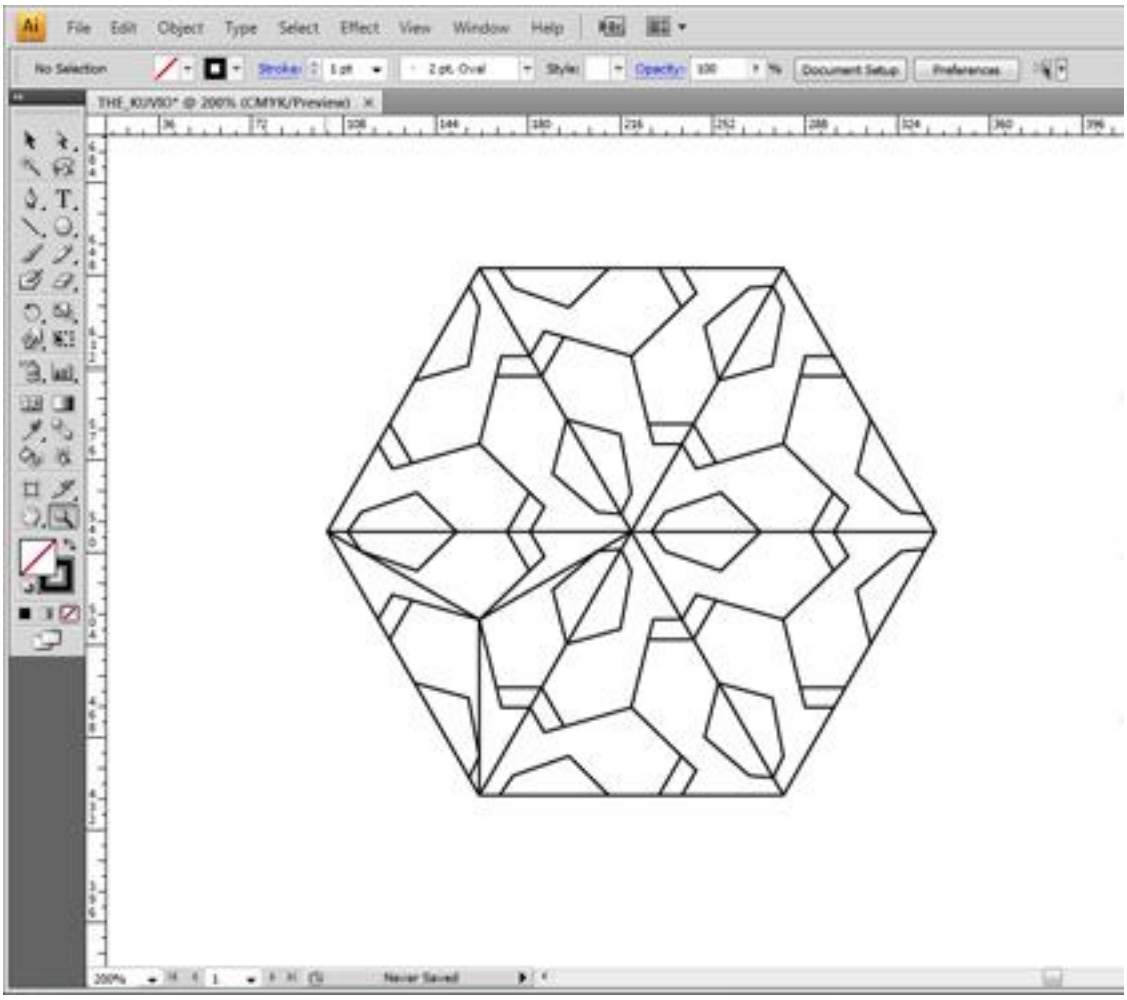
Kuvio 24. Yhteen kolmioista piirretään kuvion osa, joka käännetään sitten muihin kolmioihin.

Valitsen vektoriviivaimella piirtämäni viivat, teen niistä kopion, ja käännän sitä 120 astetta. Tämä tapahtuu valitsemalla ylävalikosta Object > Transform > Rotate... Siitä aukeaa pieni ikkuna, johon voi määritellä, kuinka monta astetta kuviota haluaa kääntää. Kun kuvio on käännetty, asetan sen paikoilleen ja toistan saman vielä viimeiseen kolmioon (kuvio 25). Tässä vaiheessa ryhmitän kuvion viivat yhdeksi ryhmäksi helpottaakseni niiden kopiointia.



Kuvio 25. Yksi kuudesosa kuviosta valmiina.

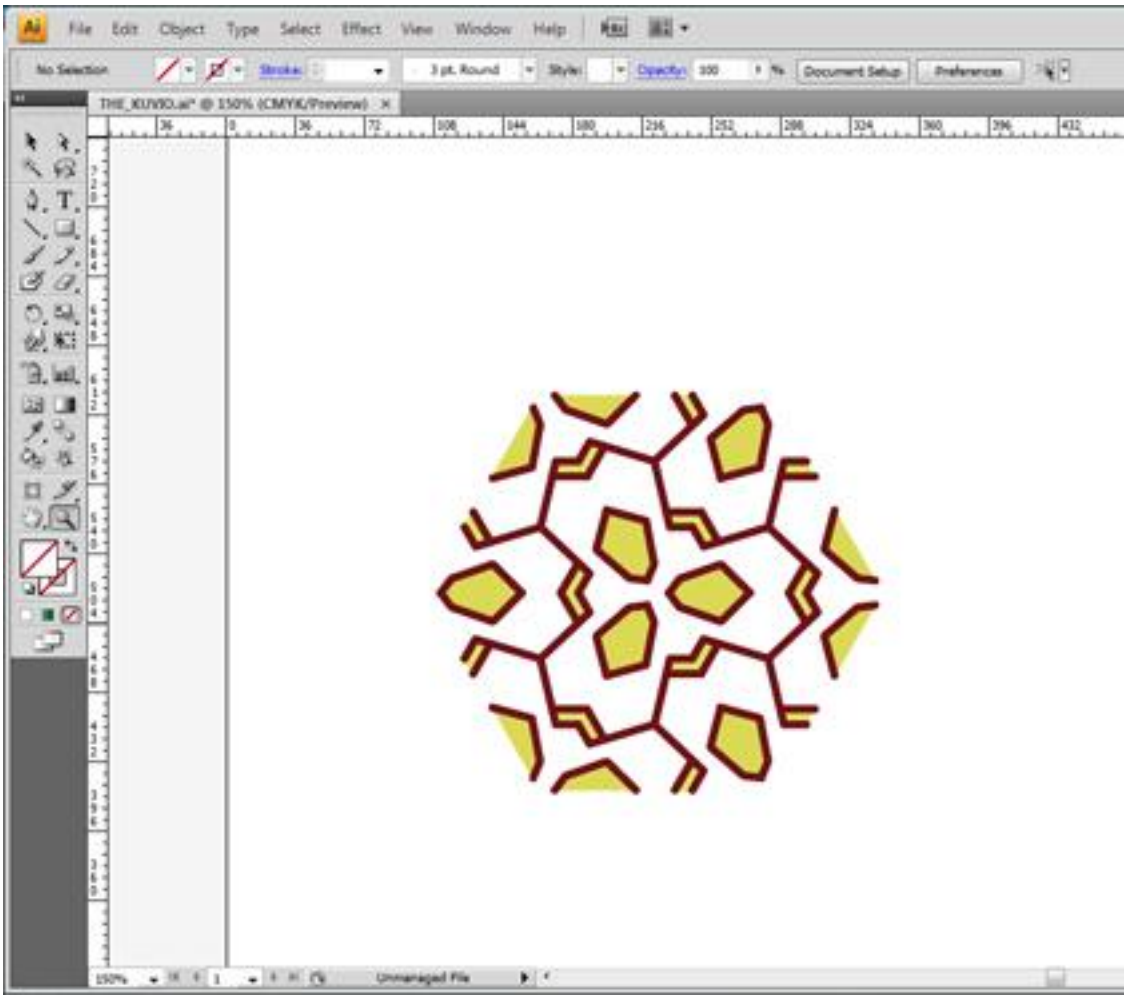
Kopioin ryhmän ja käännän sen peilikuvaksi valitsemalla ylävalikosta Object > Transform > Reflect... Ei ole väliä, peilaako kuvion horisontaalisesti vai vertikaalisti, sillä sen saa molemmista käännettyä oikein päin ja asetettua kolmioon. Nämä kaksi kolmiota toimivat parina toisilleen, ja kopioimalla ja kääntämällä ne 120 astetta saa täytettyä loputkin osat kuusikulmiosta, ja syntyy valmis kuvio (kuvio 26).



Kuvio 26. Kuusikulmio täytetty, eli kuvio on ääriviivoiltaan valmis.

Ryhmitän tämän kuvion yhdeksi ja poistan kuusikulmion ja kolmioiden viivat, sillä ne olivat vain apuviivoja.

Nyt on hyvä vaihe muokata syntyneen elementin väriä tai viivojen paksuutta. Ne on helpompi hahmottaa nyt, kun elementti on kokonainen, eikä vain 1/18 sitä. Päätin luonnoksessa pitää värit yksinkertaisena, joten valkoisen taustan lisäksi käytän vain kahta väriä: punaista, joka on RGB-arvoiltaan 110, 20, 25 ja keltaista, joka on arvoiltaan 220, 215, 85. Paksunnan samalla myös viivoja, jotta kuvio ei olisi niin paljas (Kuvio 27).

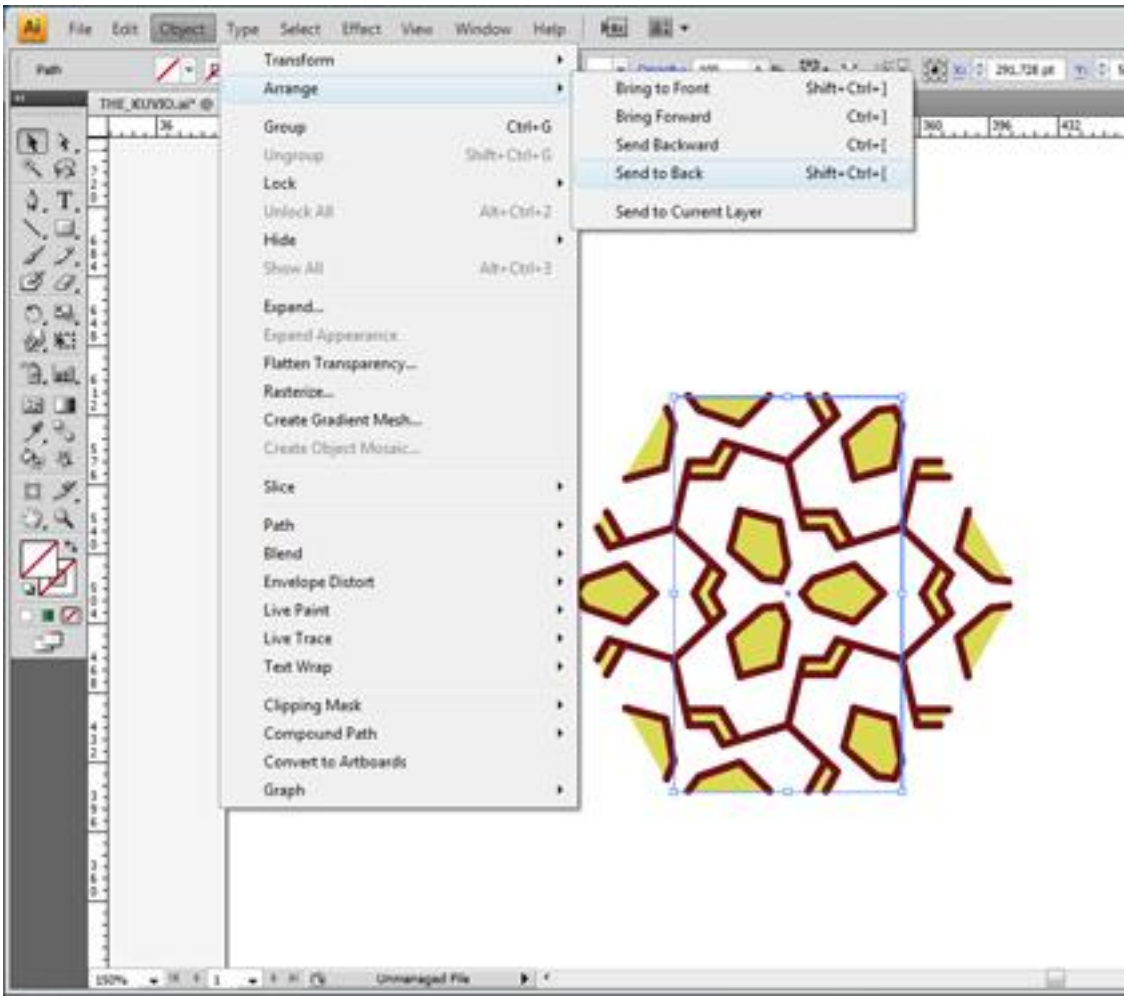


Kuvio 27. Valmis kuvio jossa on punaiset ääriviivat ja keltainen täyteväri.

Tässä vaiheessa kuvio on kaikilta reunoiltaan toistuva. Sitä voi toistaa käsin kopioimalla ja asettelemalla reunat tarkasti vastakkain. Haluan kuitenkin tehdä siitä itsestään toistuvan valmiin kuvion luomalla siitä patternin Illustratoriin. Illustrator ei osaa tehdä kuusikulmiosta tasaisesti toistuvaa, vaan kaikkien sen patternien on oltava suorakulmioita, jotta ne toistuisivat saumatta. Mikäli patterniksi tehtävä kuvio ei ole suorakulmio, Illustrator luo siitä automaattisesti sellaisen jättämällä tyhjää tilaa kuvion ympärille toistettaessa. Tästä syystä joudun tekemään vielä hieman lisää töitä, jotta kuvio toistuisi halutun lailla.

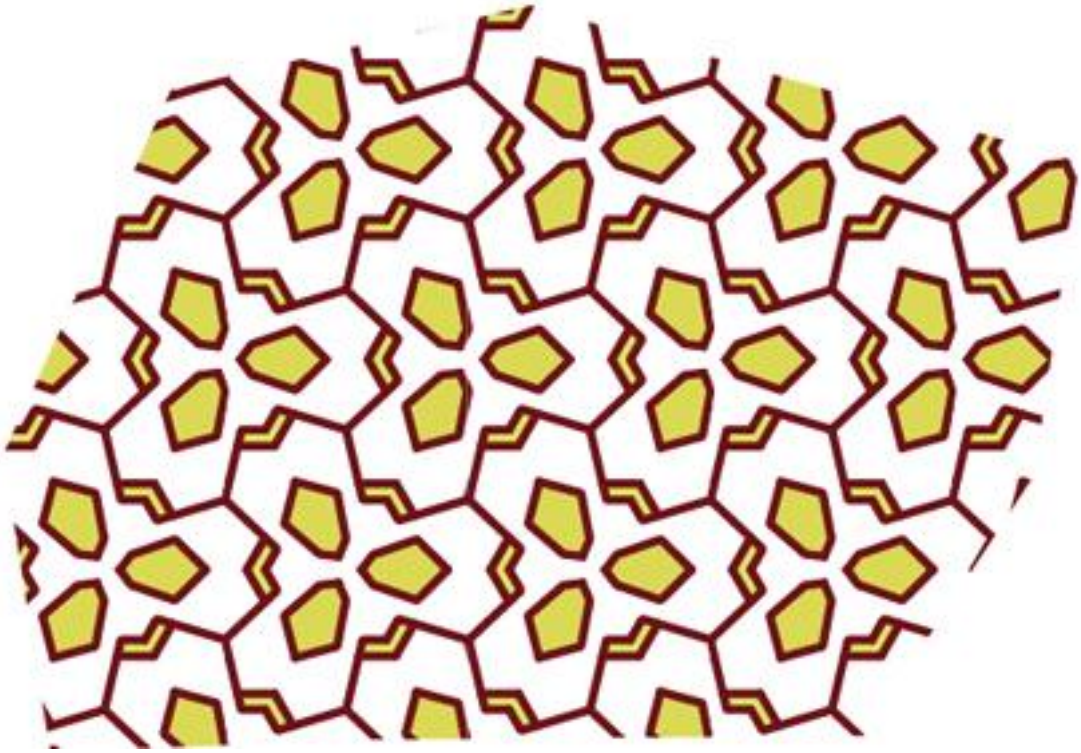
Piirrän kuvion päälle suorakulmion käyttäen Rectangle Tool –työkalua. Kun piirrän suorakulmion, varmistan, että sen sisälle jäävä alue on kaikilta reunoiltaan toistuva, eli vasen ja oikea reuna sekä ylä- ja alareuna täsmäävät viivoiltaan. Mikäli ne eivät täsmäisi, reuna-alueet tulisivat näkyviin kuvion toistuessa eikä kuvio olisi saumaton. Suo-

rakulmiolla ei ole täyttöväriä eikä reunojen väriä, jotta se ei näkyisi itse kuviossa. Siirrän suorakulmion kuvion taaimmaiseksi valitsemalla sen ja siirtymällä ylävalikosta Object > Arrange > Send to Back, esitetty kuviossa 28. Suorakulmion sisäänjäävä osa kuviota on siis se, mikä toistuu patternissa.



Kuvio 28. Kuviosta toistuu vain suorakulmion muotoinen alue, joka määritetään etukäteen.

Tehdäkseni patternin valitsen kuvion ja suorakulmion ja siirryn ylävalikosta Edit > Define Pattern... Tästä aukeaa ikkuna, jossa voi nimetä kuvion, ja painamalla OK pattern on luotu. Kokeilen valmista kuviota piirtämällä satunnaisen muotoisen alueen ja valitsemalla sen täyttöväriksi juuri tehdyn patternin (kuvio 29). Patternia voi myös skaalata pienemmäksi tai suuremmaksi, ja sillä voi täyttää minkämuotoisia alueita tahansa. Alueiden ei myöskään tarvitse olla yhtenäisiä, sillä Illustrator automaattisesti linjaa kuvion aina täsmäämään kaikkien elementtien kesken.



Kuvio 29. Kuviolla voi täyttää minkä muotoisia alueita tahansa.

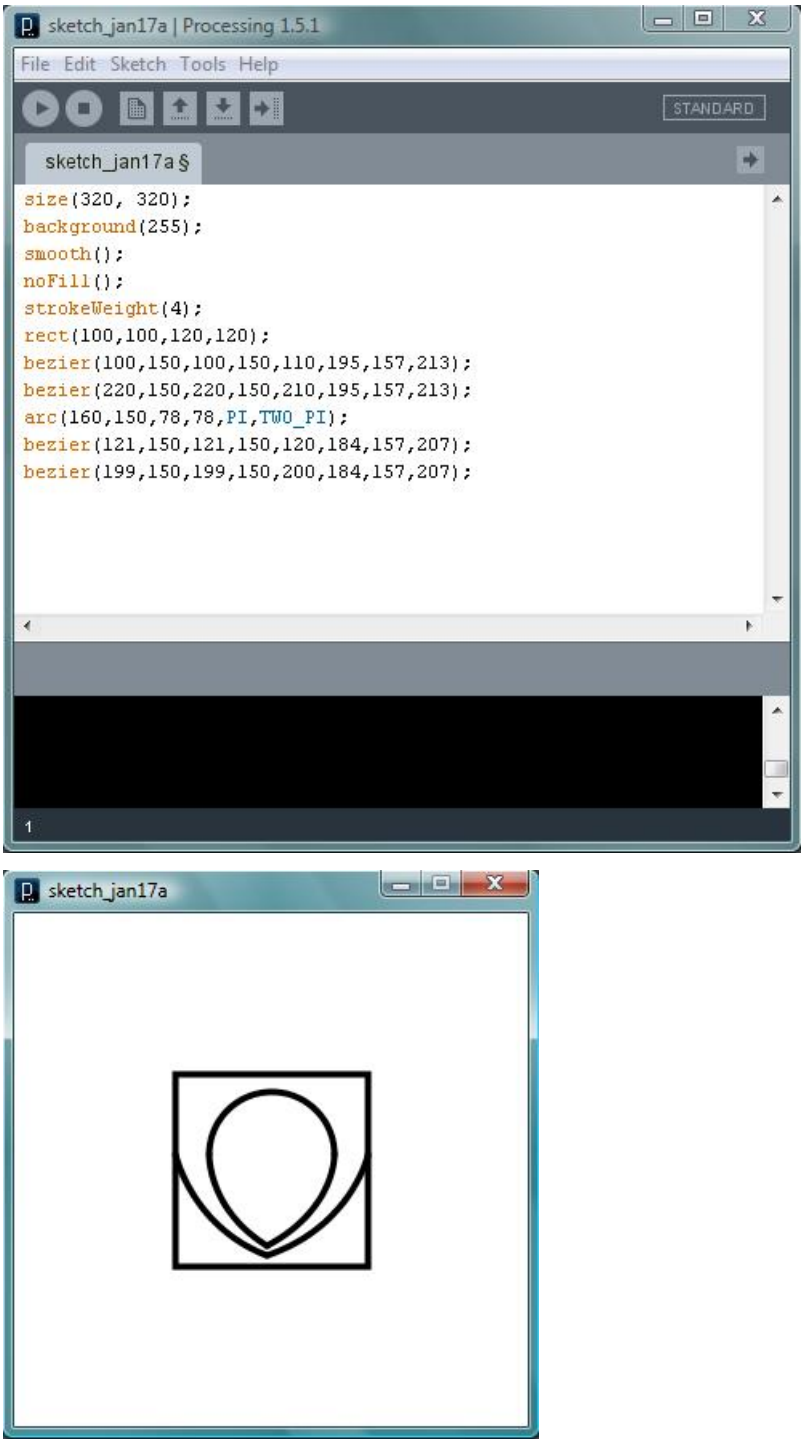
3.4 Mikä on Processing?

Casey Reas ja Ben Fry kehittivät visuaalisen ohjelmointikielen Processing vuonna 2001. He olivat samaan aikaan MIT Media Labissa, tunnetun media-alan moniosaajan John Maedan oppilaina. (Processing.org, 2011.)

Processingilla on avoin lähdekoodi, sillä sen halutaan rohkaisevan käyttäjiä sen käytössä ja olemaan mukana ohjelman parantamisessa. Sen voi ladata ilmaiseksi, jolloin esimerkiksi kouluilla ja opiskelijoilla on paremmat mahdollisuudet päästä käyttämään sitä, ja heidän osallistumisensa puolestaan auttaa Processingia kasvamaan ja kehittymään. (Reas & Fry 2007, 1.)

Processing luotiin opettamaan ohjelmoinnin perusteet visuaalisesti, olemaan luonnosvihko ohjelmille ja toimimaan tuotannon työkaluna. Se on siis ohjelmointikieli, kehitysympäristö ja opetusmenetelmä yhdessä. Siitä on tehty selkeä, jotta sillä pääsee nopeasti alkuun, mutta siinä on myös kehittyneitä ominaisuuksia haasteellisempaa ohjelmointia varten. (Reas & Fry 2007, 1.)

Processing-ohjelma on kahdessa ikkunassa. Tn tekstialue (text editor), johon koodi kirjoitetaan, ja ns. esitysikkuna (display window), jossa luonnosten (sketches) valmis tulos näkyy. Katso kuvio 30.

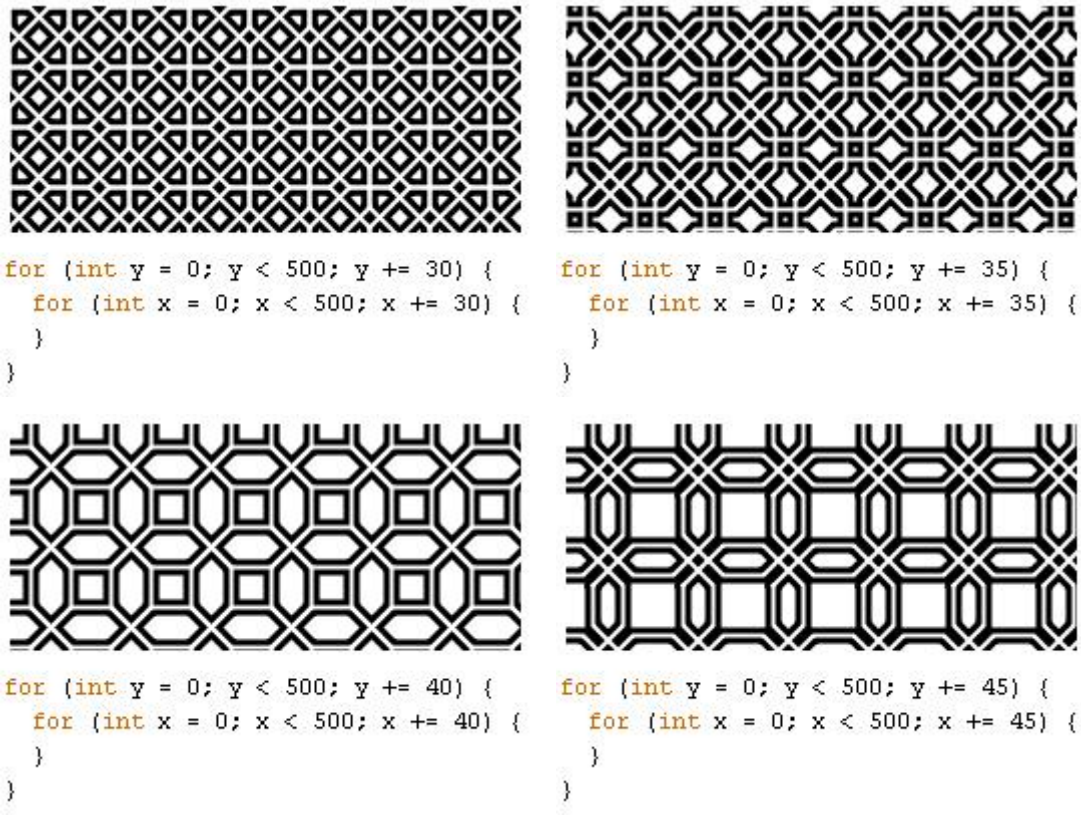


Kuvio 30. Ylempänä tekstialue, johon koodi kirjoitetaan ja alempana esitysikkuna, jossa näkyy luonnos.

3.4.1 Toteutus Processing-ohjelmointikielellä

Processing ei kielenä ole minulle entuudestaan tuttu, enkä työni yhteydessä päässyt kokeilemaan kuin hyvin rajattua osaa sen mahdollisuuksista. Totuttelin itseäni aluksi kokeilemalla vain yksinkertaisten muotojen piirtämistä ja niiden muokkaamista värien tai viivan paksuutta muuttamalla. Tein oikeastaan samaa, mitä pystyin Illustratorissa tekemään vektoreilla.

Käytän Processingia ensimmäistä kertaa, joten kieli, jota käytän on hyvin yksinkertaista ja kehitettävissä. Mutta juuri se on Processingin hyvä puoli: sillä pääsee nopeasti liikkeelle ja vaikka tekisi vain yksinkertaista koodia, silläkin saa paljon aikaan. Edistyneempää koodia tehtäessä Processingissa tarvitsee hieman matemaattista taitoa tai hahmottamiskykyä. Olen suorittanut aikoinaan pitkän matematiikan kurssit lukiossa, mutta sen jälkeen en ole matematiikan kanssa erityisemmin ollut tekemisissä. Jos muistaisin hieman paremmin perusteet, se rohkaisisi käyttämään enemmän algoritmeja mahdollistamaan hienompia lopputuloksia. Tulin kuitenkin siihen tulokseen, että vaikka ne tutuilta näyttivät, niiden uudelleen hahmottamiseen kuluisi sen verran aikaa, että pitäydyin yksinkertaisessa koodissa. Silläkin sai paljon aikaan. Piirtämällä esimerkiksi kahdeksankulmion ja lisäämällä siihen toistokäskyn syntyi muutamassa minuutissa koko pinnan kattava hieno kuvio. Säätämällä hieman toiston etäisyyttä sai aikaan näkyvän muutoksen, kuten alla olevasta kuviosta 31 näkee.



```
for (int y = 0; y < 500; y += 30) {
  for (int x = 0; x < 500; x += 30) {
  }
}
```

```
for (int y = 0; y < 500; y += 35) {
  for (int x = 0; x < 500; x += 35) {
  }
}
```

```
for (int y = 0; y < 500; y += 40) {
  for (int x = 0; x < 500; x += 40) {
  }
}
```

```
for (int y = 0; y < 500; y += 45) {
  for (int x = 0; x < 500; x += 45) {
  }
}
```

Kuvio 31. Kahdeksankulmio toistoväleillä 30, 35, 40 ja 45. Ero koodissa on pieni, mutta kuviossa huomattava.

3.4.2 Työvaiheet

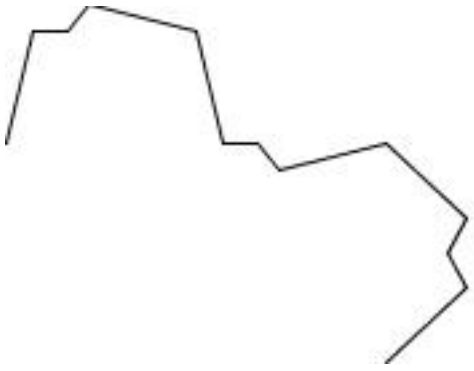
Aloitin määrittämällä tekstialueeseen luonnoksen koon size()-funktiolla, 600 x 600 pistettä. Ajattelin että 600 pistettä on sopiva koko, sillä siinä näkyy jo kuvion toistuvuus, mutta alue ei ole tarpeettoman iso. Alueen koolla ei sinänsä ole mitään väliä, sillä sitä voi muuttaa missä vaiheessa tahansa, jos on tarvetta. Määritettäessä arvoja ensimmäinen numerosarja tarkoittaa aina x-akselin arvoja, eli leveyttä tai etäisyyttä vasemmalta oikealle, ja toinen numerosarja tarkoittaa y-akselin koordinaatteja, eli korkeutta tai etäisyyttä yläreunasta alas, riippuen missä yhteydessä niitä käytetään. Asetin taustavärin valkoiseksi eli RGB-väriarvon 255. Koska valkoisessa kaikki arvot ovat 255, riittää, että sen kirjoittaa vain kerran. Seuraavaksi määritin viivan pehmeäksi smooth()-funktiolla. Mikäli tätä ei määritä erikseen, piirrettävät viivat ovat karkeita, eli pikselimäisiä. Nämä funktiot arvoineen on esitetty kuviossa 32.

```
size(600,600);
background(255);
smooth();
```

Kuvio 32. Koko, taustaväri ja viiva määritetty

Nyt kun asetukset olivat valmiina, aloin piirtää ensimmäistä osaa kuviosta. Päätin etukäteen, että minun ei ole järkevää lähestyä sen piirtämistä samalla tavalla kuin Illustratorissa, sillä siinä tulisi turhaa työtä. Sen sijaan koska Processingissa toisto komento ei ole rajoittunut suorakulmion muotoiseen alueeseen, vaikka toisto tapahtuu y- ja x-akseleille uskollisesti, voisi muoto olla mikä vain. Välttääkseni päällekkäisiä viivoja päätin, etten piirrä umpinaista muotoa, kuten Illustratorissa, vaan vain viivan, joka riittää kuvion saumattomaan toistoon. Piirsin viivan käyttämällä verteksipisteitä (vertex points). Vertex()-funktioon täytyy aina määrittää x- ja y-arvot asettamaan pisteet paikoilleen. Nämä verteksipisteet puolestaan kirjoitetaan beginShape()- ja endShape()-funktioiden sisään. Koodi tähän mennessä ja miltä se näyttää käytännössä näkyy kuviossa 33.

```
size(600,600);
background(255);
smooth();
beginShape();
vertex(0,52);
vertex(10,10);
vertex(23,10);
vertex(31,0);
vertex(71,10);
vertex(81,52);
vertex(94,52);
vertex(102,62);
vertex(142,52);
vertex(172,80);
vertex(165,93);
vertex(172,106);
vertex(142,134);
endShape();
```



Kuvio 33. Tähän mennessä piirretty kuvio

Mikäli muoto olisi suljettu, eli alkaa ja loppuu samaan pisteeseen, kirjoitettaisiin endShape(CLOSE). Jätin omani kuitenkin auki, joten pelkkä endShape() riittää. X- ja y-

koordinaatit katsoin suunnilleen luonnoksesta ja laskin sitten tarkasti niin, että ne muodostavat kuviosta symmetrisen.

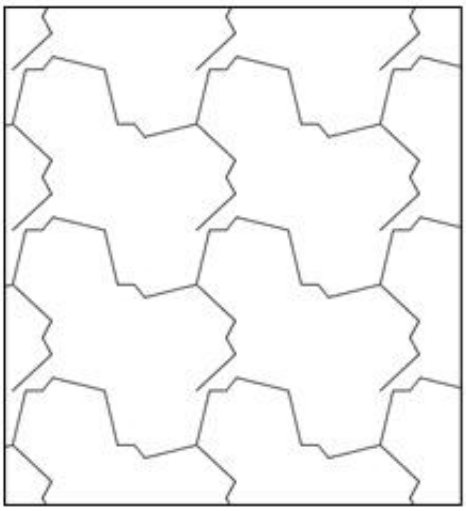
Laitoin kuvion tässä vaiheessa toistumaan, jotta näin, osuvatko viivat kuten suunnittelin. Toistettaessa käytetään for-rakennetta, joka tekee laskelmat toistoille (Reas & Fry 2007, 63). Kuviossa 34 näkyy koodin pätkä, joka tähän vaaditaan.

```
for (int y = -54; y < 600; y += 124) {
  for (int x = -30; x < 600; x += 142) {
  }
}
```

Kuvio 34. Koodi toistolle

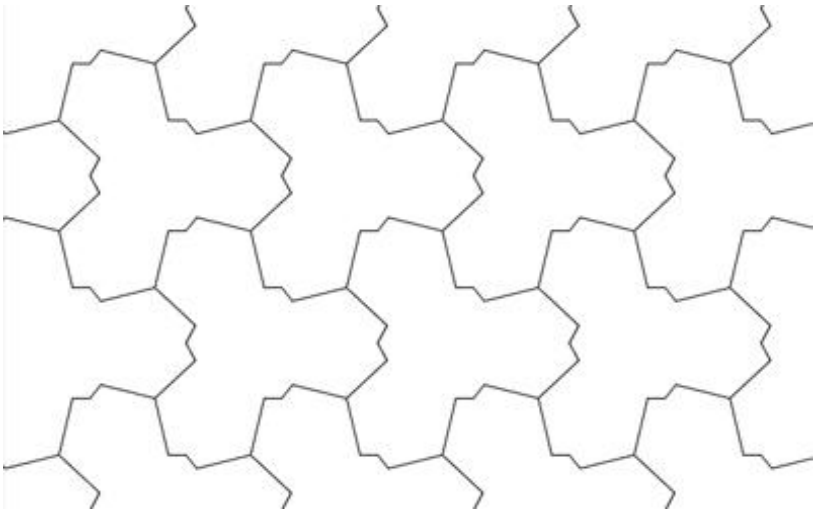
For-rakennetta seuraa suluissa kolme tekijää: ensinmäisenä int, joka määrittää alkuperäisen pisteen, josta lähdetään liikkeelle. Tässä tapauksessa int y = -54, eli kuvio lähtee 54 pistettä ylärajan yläpuolelta, siksi negatiivinen luku. Y < 600 tarkoittaa, että kuvio toistuu, kun arvot ovat pienemmät kuin 600 ja arvojen ylittäessä tämän se lakkaa toistumasta. Tämä ei kuitenkaan takaa kuviolle tasaista reunaa, sillä kuvio toistuu aina kokonaisena. Jos kuvio alkaa ennen määriteltyä rajaa, toistuu se sen yli kokonaisena. Määritin oman kuvion toistolle ylärajaksi 600, koska se on joka tapauksessa valmiin luonnoksen koko. Viimeinen tekijä määrittää, millä etäisyydellä kuvio toistuu, eli kuinka kauas tai lähelle seuraava kuvio tulee. Plus- tai miinus-merkki määrittää suunnan, mihin seuraava kuvio siirtyy. Omassa esimerkissäni y += 124 tarkoittaa, että seuraava kuvio alkaa 124 pistettä edeltävän kuvion alun alapuolella. Koska kuvioni korkeus on 124 ja toisto tapahtuu 124 pisteen välein, seuraava kuvio alkaa heti siitä, mihin edellinen loppuu. Kaikki tämän toiston piiriin kuuluva tapahtuu { ja } –merkkien sisällä. { -merkki avaa toistoalueen ja } -merkki lopettaa sen. Joudun nyt muuntamaan vertekseille määritetyt arvot x- ja y-arvoiksi. Tämä käy lisäämällä x+ ja y+ olemassa olevien arvojen eteen. Kuvio 35.

```
size(600,600);
background(255);
smooth();
for (int y = -54; y < 600; y += 124) {
  for (int x = -30; x < 600; x += 142) {
beginShape();
vertex(x,y+52);
vertex(x+10,y+10);
vertex(x+23,y+10);
vertex(x+31,y);
vertex(x+71,y+10);
vertex(x+81,y+52);
vertex(x+94,y+52);
vertex(x+102,y+62);
vertex(x+142,y+52);
vertex(x+172,y+80);
vertex(x+165,y+93);
vertex(x+172,y+106);
vertex(x+142,y+134);
endShape();
  }
}
```



Kuvio 35. For-rakenne on lisätty piirretyn verteksin eteen ja vieressä miltä kuvio näyttää tässä vaiheessa.

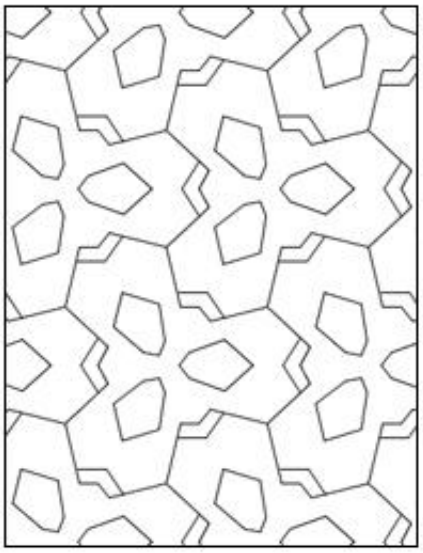
Näin, että vaakatasossa kuvio toistui oikein, eli oikeat pisteet osuivat kohdakkain. Pystysuunnassa ne eivät kuitenkaan täsmänneet. Tämä johtuu siitä, että tässä muodossa seuraava osa ei tule suoraan edeltävän alle, vaan puolet kuvion leveydestä sivuun. Tästä syystä jouduin tekemään toisen toiston alkamaan puolen kuvion verran aikaisemmin vasemmalta. Lisäksi muutin y-akselin toiston arvot niin, että kuviot toistuvat joka toisessa rivissä, jolloin ne menevät limittäin. Valmis kuvio näkyy kuviossa 36. Kaikki viivat osuvat kohdalleen muodostaen suljetun alueen.



Kuvio 36. Kuvio toistuu saumattomasti

Seuraavaksi siirryin täyttämään kuviot niihin tulevilla viisikulmioilla ja viivoilla. Viisikulmioihin käytin samaa beginShape()- ja endShape()-funktiota kuin aiemminkin, mutta nyt, kun kyseessä on suljettu muoto, kirjoitin endShape(CLOSE). Tein näitä viisikulmioita kolme. Line() funktiota käyttäen piirsin viivat, jotka tulevat suljetun alueen kärkiin. Line()-funktioon määritetään ensin alkupisteen x- ja y-arvot, ja viereen loppupisteen x- ja y-arvot, kuten kuviossa x näkyy. Sain x- ja y-arvot arvioimalla ja siirtelemällä pisteitä, kunnes ne olivat symmetrisesti. Myös näiden arvojen eteen on lisätty x+ ja y+ -merkinnät, jotta niihin pätee sama toisto kuin muihinkin pisteisiin. (Kuvio 37)

```
line(x+23, y+112, x+11, y+93);
line(x+11, y+93, x+23, y+74);
line(x+8, y+20, x+28, y+20);
line(x+28, y+20, x+40, y+3);
line(x+80, y+42, x+100, y+42);
line(x+100, y+42, x+112, y+60);
fill(225,235,90);
beginShape();
vertex(x+34, y+66);
vertex(x+40, y+42);
vertex(x+66, y+50);
vertex(x+70, y+76);
vertex(x+66, y+86);
vertex(x+55, y+84);
endShape(CLOSE);
beginShape();
vertex(x+34, y+120);
vertex(x+55, y+104);
vertex(x+66, y+102);
vertex(x+70, y+112);
vertex(x+66, y+138);
vertex(x+40, y+146);
endShape(CLOSE);
beginShape();
vertex(x+80, y+93);
vertex(x+87, y+84);
vertex(x+112, y+75);
vertex(x+132, y+93);
vertex(x+112, y+111);
vertex(x+87, y+102);
endShape(CLOSE);
```

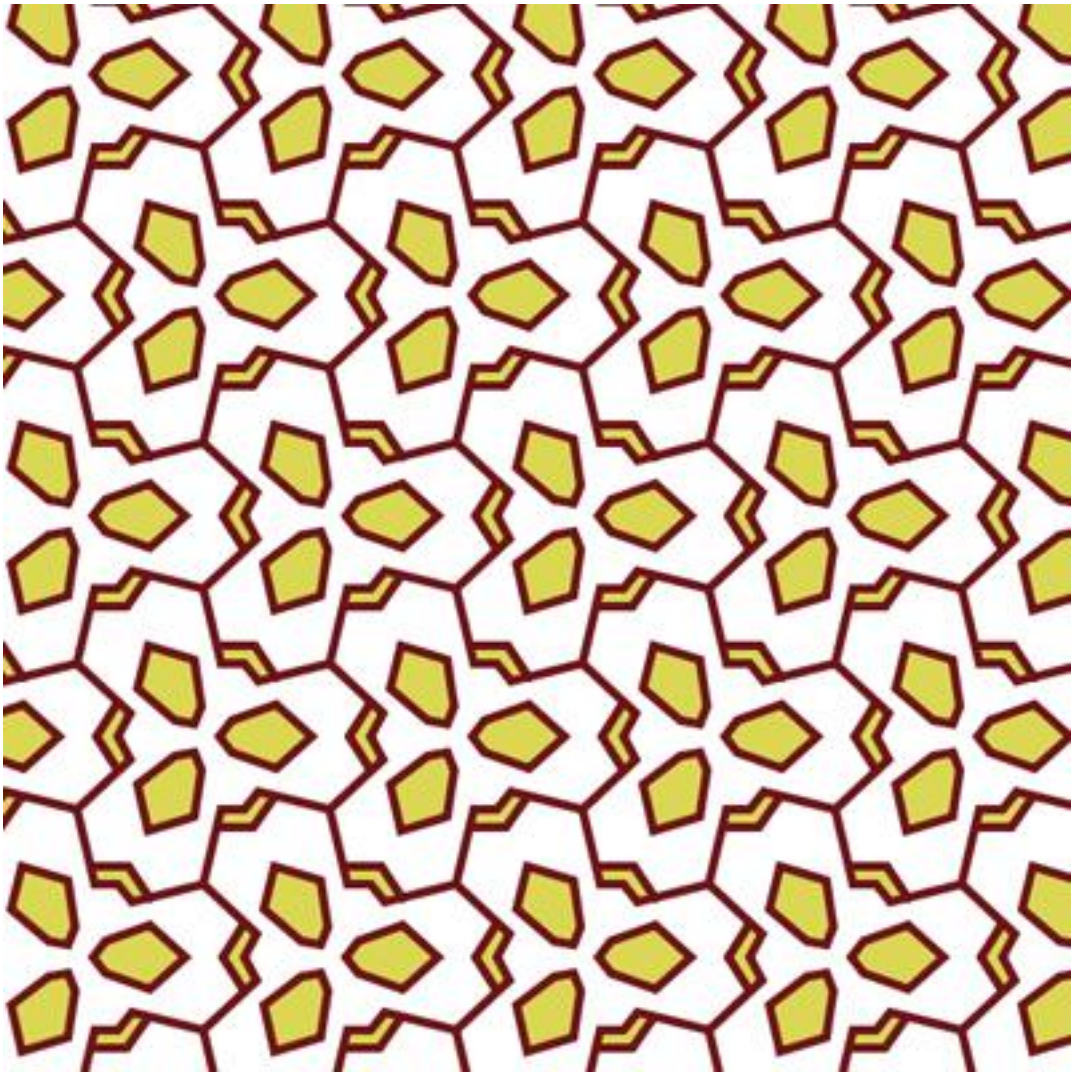


Kuvio 37. Koodi viisikulmioita ja viivoja varten.

Viimeisenä vaiheena muokkasin vielä ulkoasua, sillä nyt se koostui vain mustista viivoista. Muutin viivan paksuutta strokeWeight()-funktiolla ja paksuudeksi 5. Stroke()-funktiolla saa määritettyä viivalle värin, tässä tapauksessa RGB-arvot 110, 20, 25, eli sama punainen kuin Illustratorissakin.

Koodissa alimmaksi eli järjestyksessä viimeiseksi kirjoitettu koodi tulee näkymään päällimmäisenä. Halusin värittää kärkiin jäävät alueet keltaisiksi, mutta ne eivät olleet suljettu muoto, joten piirsin niiden alle erillisen alueen, joka on keltainen. Siirsin kirjoitetun koodin ensimmäiseksi, jolloin se tuli näkyviin kaikkein alimmaksi. Viisikulmiot ovat puolestaan suljettu alue, joten niiden sisälllön värin pystyin muuttamaan fill()-

funktiolla. Käytän samaa keltaista kuin Illustratorissakin, jonka RGB-väriarvo on 220, 215, 85. Tämä aiheuttaa sen, että alempana koodissa olevaan kuvioon pätee myös tämä fill()-funktio, mitä en halunnut, sillä osissa sitä kuuluu olla vain ääriviivat. Kirjoitin siksi sen koodin eteen noFill()-funktion, joka poistaa mahdolliset täytevärit. Jouduin sitten jälleen toistamaan fill()-funktion alempien viisikulmioiden kanssa. Nyt, kun kuvio toistuu ja sen ulkoasu on kohdallaan, kuvio on viimein täysin valmis. Valmis kuvio näkyy kuvioissa 38. Koodi on kokonaisuudessaan Liitteessä 1.



Kuvio 38. Valmis kuvio

3.5 Toteutusten vertailua

Molemmilla ohjelmilla tehdyistä toteutuksista tuli samanlaiset, kuten oli tarkoitus. Molemmat ohjelmat ovat siis yksinkertaisilta visuaalisilta tekijöiltään yhdenvertaiset. Mikäli toteuttamani kuvio olisi ollut monimutkaisempi, olisivat varmasti näiden ohjelmien erot tulleet paremmin esiin lopputuloksissakin.

Tehdessä kuitenkin näkee selvästi, kuinka täysin erilailla näillä ohjelmilla tulee kuvion rakentamista lähestyä. Siinä, missä Illustratorilla on helppo luoda suoraan kuvion yksittäinen elementti ja rajata sitä sitten toistoa varten, on se Processingissa haasteellisempaa. Vektoreilla piirrettäessä asetetaan ankkuripisteitä, ja samoin Processingissa piirtämällä asetin tiettyjä pisteitä, joiden väliin viivat piirtyivät. Erona oli kuitenkin se, että jouduin itse laskemaan pisteiden koordinaatit, kun Illustratorissa vektoreilla piirrettäessä ne pystyi vain asettamaan valitsemaansa kohtaan. Molemmissa oli yhtä paljon mahdollisuuksia tehdä virhe tarkkuuden suhteen pisteen sijoituksessa. Illustratorissa, koska pisteet asetetaan vapaalla kädellä ja Processingissa, koska pisteiden sijainteja laskiessa saattaa tulla vahingossa laskuvirhe.

Processingissa kuitenkin itse toisto on yksinkertaisempi, sillä siihen tarvitaan vain yksi komento ja se on nopeasti muutettavissa ja testailtavissa, jos on tarve. Illustratorissa kuvio pitää aina ensin määrittää patterniksi, että sitä pääsee kokeilemaan, ja sitten jos muutettavaa on, pattern on määritettävä taas uudestaan tehtyjen muutosten jälkeen.

Processingilla saa myös tarkemmin tehtyä juuri minun kuviossani tarvittavat käännökset ja kopiot. Illustratorissa sain kyllä käännettyä niitä tarkassa kulmassa, mutta paikoilleen jouduin asettamaan ne käsin. Tässä toki avustivat olemassa olevat ankkuripisteet, joihin osat liittää, mutta silti käsin aseteltaessa voi pientä virhettä tapahtua. Processingilla nämä kaikki asetukset määritellään tiettyihin arvoihin, jolloin ne asettuvat tarkasti paikoilleen.

Itselläni Processingilla piirtämistä helpotti kuitenkin huomattavasti se, että olin ensin tehnyt kuvion Illustratorilla. Näin Processingilla tehdessä minulla oli jotain, mistä ottaa mallia. Ilman mallia koordinaattien hahmottaminen ja laskeminen olisi ollut huomattavasti hitaampaa ja työläämpää.

Ihanteellinen tilanne olisikin näiden kahden yhdistäminen. Yksittäisen kuvion voisi esimerkiksi piirtää Illustratorissa ja tuoda sitten Processingiin, jossa voisi tehdä tarpeelliset toistot. Tämä helpottaisi esimerkiksi tilanteissa, jossa kuvion halutaan toistuvan päällekkäin edeltävän kanssa. Vastaavasti Processingilla voisi tehdä tarkasti pohjan kuviolle, ja tämän pohjan voisi sitten viedä Illustratoriin, jossa sen voisi vapaammin täyttää. Esimerkiksi epäsäännöllisten, vapaiden muotojen piirtäminen Processingilla pisteiden koordinaatteja laskien olisi hyvin tuskallista.

3.6 Processing-ohjelmointikielen ja Illustratorin yhdistäminen

Näiden ohjelmien yhdistäminen on täysin mahdollista, ja kokeilin sitä aluksi yksinkertaisella esimerkki kuviolla. Piirsin ensin toistettavan elementin Illustratorissa (kuvio 39) ja talletin sen PNG-tiedostoksi. Valitsin tiedostotyypiksi PNG:n, koska se säilyttää mahdollisen läpinäkyvyyden, eikä korvaa sitä esimerkiksi valkoisella taustavärillä kuten JPEG. PNG-tiedostotyyppi tallettaa yhtä paljon värejä kuin JPEG joten läpinäkyvyyden säilyminen ratkaisi näiden välillä. (Reas & Fry 2007, 96.)

Kuvio 39. Illustratorissa tehty elementti Processingissa toistoa varten

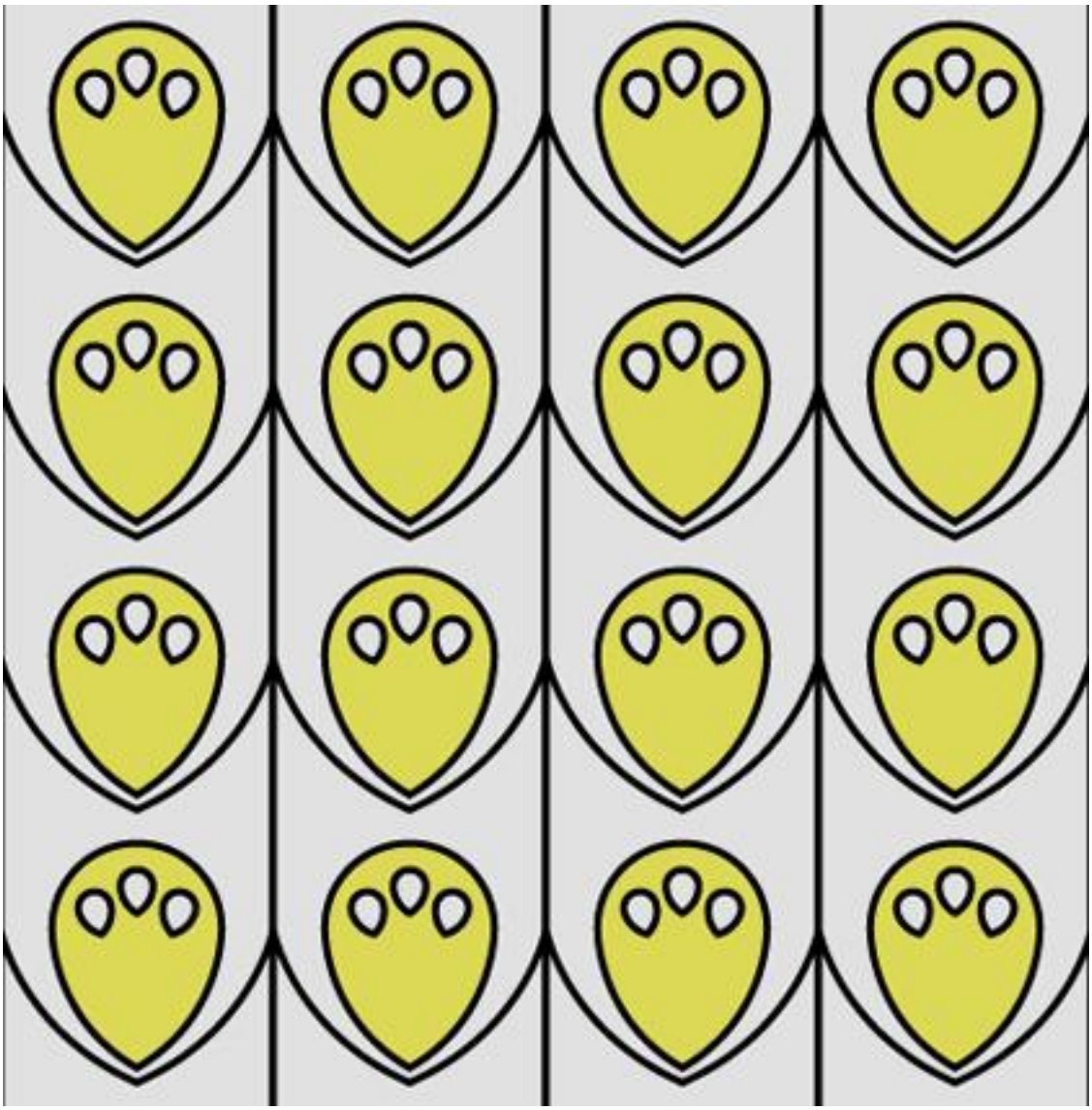
Seuraavaksi kirjoitin Processingissa koodin, mikä koskee kuvaa. Kuvia varten oleva datatyyppi on Pimage, joka määritetään alkuun. Seuraavaksi kuva tuodaan luonnokseen loadImage()-funktiolla, jossa sulkuihin kirjoitetaan ” ”-merkkien sisään tuotavan tiedoston nimi ja tiedostotyyppi, tässä tapauksessa illupros.png. Jotta kuva latautuu luonnokseen, täytyy se tallettaa luonnoksen data kansioon. Tämä tapahtuu valitsemalla ylävalikosta Sketch > Add File... ja etsimällä ja valitsemalla oikean kuvan. Viimeiseksi image()-funktiolla voi vielä määrittää kuvan sijainnin x- ja y-akseleilla, sekä määrittää sen leveyden ja korkeuden, mikäli ei halua säilyttää alkuperäisiä. Jos leveyden ja kor-

keuden jättää määrittämättä, tulee kuvio sen kokoisena kuin se on alunperin talletettu. Nämä on kaikki esitelty koodin pätkässä kuviossa 40.

```
size (600,600);
background(225);
for (int y = 0; y < 600; y += 113 ) {
  for (int x = 0; x < 600; x += 113) {
PImage img;
img = loadImage("illupros.png");
image(img, x, y);
  }
}
```

Kuvio 40. Vaadittavat funktiot kuvan lataamiseksi luonnokseen

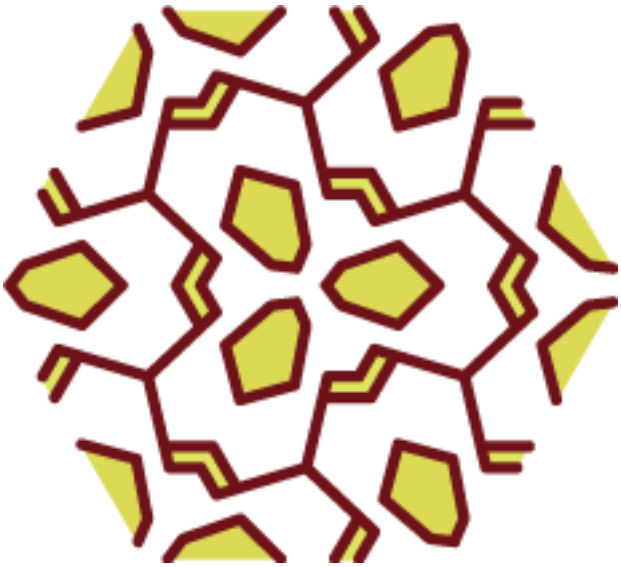
Lähtökohtaisesti olin määrittänyt koko luonnoksen kooksi 600x600 pistettä ja taustaväriksi vaaleanharmaan. Vaaleanharmaa taustaväri tulee esille, sillä Illustratorissa tekemäni elementti on mustia viivoja ja keltaista täyteväriä lukuunottamatta läpinäkyvä. Viimeiseksi lisäsin koodiin vielä toiston, joka on selitetty vaihe vaiheelta kappaleessa 3.4.2. Työvaiheet. Tällä lyhyellä koodin pätkällä valmiiksi tehty elementti toistuu saumattomasti (kuvio 41).



Kuvio 41. Illustratorissa tehty osittain läpinäkyvä elementti toistettuna Processingilla harmaalle taustalle.

Koodi, jolla sain toistuvan kuviopinnan on huomattavasti lyhyempi ja helpompi, kuin aiemmin täysin Processingilla tekemäni kuvion koodi. Tekemällä kuvion vektorigrafiikkaohjelmalla ja tuomalla sen Processingiin säästää huomattavasti vaivaa ja saa paljon nopeammin kirjavampia muotoja. Koska kuvan tuominen Processingiin ja toistaminen sillä oli niin helppoa, päätin kokeilla sitä vielä lopullisella kuviollani.

Tein Illustratorissa toiston rajaamalla kuusikulmiosta suorakulmion muotoisen alueen. Nyt päätin kuitenkin tehdä toiston käyttämällä alkuperäistä kuusikulmiomuotoa. Talletin siis kappaleessa 3.3.2 Työvaiheet tekemäni väritetyn kuusikulmion PNG-tiedostoksi (kuvio 42).



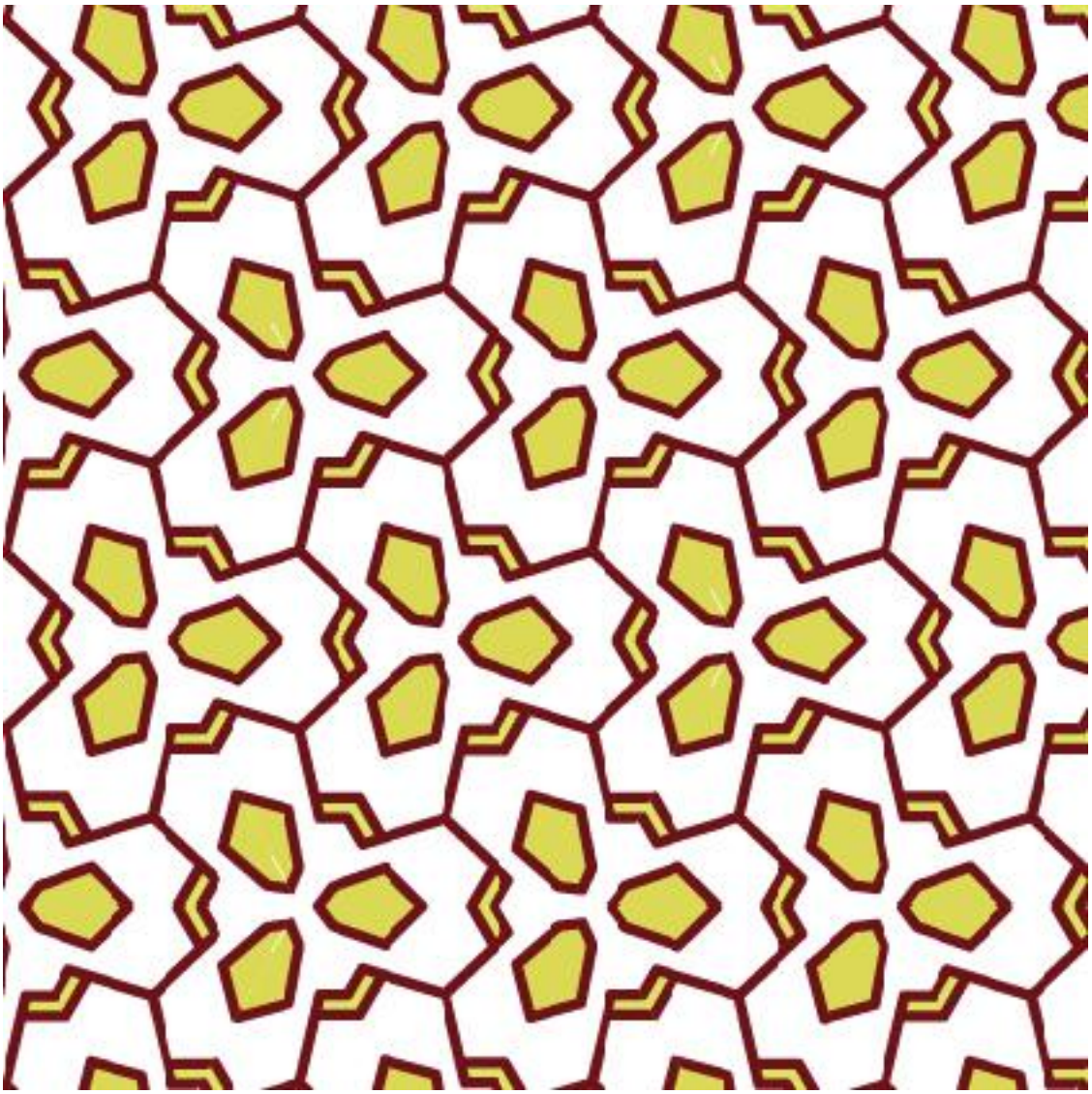
Kuvio 42. Kuvio kuusikulmion muotoisena talletettu PNG-tiedostoksi

Sitten tein Processingissa samanlaisen pohjan kuin aikeisemminkin, eli kooksi 600x600 pistettä ja valkoinen tausta. Asetin toistossa x- ja y-akselin toiston alkamaan hieman ennen luonnoksen vasenta reunaa ja yläreunaa, sillä muuten sinne jäisi valkoista. Tämä johtuisi siitä, että kuusikulmio ei toistu suoraan kuutiotoistona, vaan kuusikulmiot menevät hieman limittäin, jotta ne toistuisivat saumattomasti. Tästä samasta syystä joudun määrittämään kaksi toistoa, joista toinen kulkee hieman alempana ja hieman sivummassa kuin toinen. (Kuvio 43).

```
size (600,600);
background(255);
for (int y = -106; y < 600; y += 206 ) {
  for (int x = -50; x < 600; x += 340) {
PImage img;
img = loadImage("THE_KUVIO.png");
image(img, x, y, 226, 206);
  }
}
for (int y = -3; y < 600; y += 206 ) {
  for (int x = 120; x < 600; x += 340) {
PImage img;
img = loadImage("THE_KUVIO.png");
image(img, x, y, 226, 206);
  }
}
```

Kuvio 43. Koodissa kuusikulmiota toistetaan kahdessa rivissä, jotta se toistuu saumattomasti

Kuusikulmiot toistuivat kuten suunnittelin. Niistä tuli paljon nopeammin saumattomasti toistuva pinta, kun kummallakaan, Illustratorilla tai Processingilla, yksin. (Kuvio 44).



Kuvio 44. Illustratorin ja Processing-ohjelmointikielen yhdistelmällä tehty saumattomasti toistuva kuvio

Halusin nimenomaan tehdä toiston käyttäen kuusikulmiota, sillä en käyttänyt sitä kummassakaan, Illustratorissa tai Processingissa, yksin toistettaessa. Jälkeenpäin ymmärsin kuitenkin, että olisi ollut järkevämpää tehdä kuten Illustratorissa ja toistaa kuusikulmiosta leikattua suorakulmiota. Ensinnäkin toistaminen olisi entistä helpompaa, sillä sen voisi tehdä vain yhdellä toistokäskyllä. Suorakulmion muoto on helppo saada tarkasti suoraksi, ja koska se toistuu suoraan kiinni edelliseen, ei sen toistoväliä tarvitse

hakea. Siihen riittää, että määrittää y-akselin toistoväliksi kuvion korkeuden, ja x-akselin toistoväliksi kuvion leveyden.

Kun käytin kuusikulmiota toistamiseen, huomasin sitä toistaessa, että kuusikulmio ei ollut tarkkaan oikean muotoinen. Tämän näkee jos kuviota 44 katsoo tarkemmin. Jouduin peittelemään tätä pientä virhettä säätämällä toistoväliä. Sen takia kuviossa on joissain kohdin pientä päällekkäisyyttä ja toisissa kohdissa on pieni rako. Yksittäisestä elementistä ei huomaa paljain silmin, että se ei ole tarkkaan oikean muotoinen, mutta kun kuvio toistuu ja elementtien pitäisi loksahtaa saumattomasti paikoilleen, tulevat nämä virheet esiin.

Tällä tavalla tehden on toistamistapa periaatteessa aidoin, eli ei oteta kuusikulmion sisältöä huomioon, vaan vain sen ulkomuoto.

4 Yhteenveto

Työssä suunnittelin ja toteutin kuvion, tutkittuani ensin sen teoriaa. Toteuttamani kuvio toistuu jaksollisesti ja saumattomasti, kuten oli tarkoitukseni. Ensimmäisen toteutuksen tein Adobe Illustratorilla, joka oli minulle entuudestaan tuttu. Toisen toteutuksen tein Processing-ohjelmointikielellä, ja sitä varten jouduin hieman harjoittelemaan etukäteen, että opin ymmärtämään ja kirjoittamaan koodia.

Minulla kesti kauan päästä työssä alkuun, sillä jämähdin luonnosteluvaiheeseen. Uskon, että minun oli niin vaikea päästä alulle kuvion kanssa, koska minulla ei ole aiempaa kokemusta tällaisen suunnittelusta, mutta kuitenkin tavoitteita ja odotuksia paljon. En onnistunut millään luonnostelemaan kuviota, joka olisi täyttänyt tavoitteeni, ja jota olisin halunnut viedä eteenpäin. Halusin sisällyttää kuvioon välttämättä joitain tiettyjä toistuvuussääntöjä, vaikka olisin voinut aloittaa luomalla aluksi vain yksinkertaisen kuvion ja lähteä siitä. Lisäksi minulla ei ollut kunnollista mielikuvaa, minkä näköisen kuvion haluaisin, joten en tiennyt, mihin suuntaan työstää mahdollisia pohjia. Jouduin pakottamaan itseni luopumaan tavoitteistani ja vaatimuksistani jossain vaiheessa, kun ne rajoittivat työni etenemistä liikaa. Päätin, etten aseta kuviolle etukäteen mitään tiettyä käyttötarkoitusta, sillä se tuntui eniten rajoittavan suunnittelua.

Tämän jumiutumisen myötä koko työn edistyminen pysähtyi täysin sillä koin, etten voi jatkaa mitään osaa työstä ennen kuin tiedän, millaisen kuvion haluan tehdä. Tätä vaihetta kesti useampi kuukausi ja työn eteneminen vain mateli kasvavasta kiireestä huolimatta. Hyvä puoli tässä oli, että ajatukset työstä selkiintyivät hieman siinä samalla. Ongelmia oli nimittäin alussa myös työn rajauksen kanssa. En osannut päättää miten lähestyisin kuvion toteutusta, miten paljon määräisi sen tuleva käyttötarkoitus? Suunnittelisinko kuvioita yhden vai useamman, esimerkiksi sarjan samantyyppisiä kuvioita? Mitä tutkisin kuvioista, rakennetta, eroja aikakausittain tai kulttuurityypeittäin? Kun lopulta sain määriteltyä rajauksen ja keksin, millaisen kuvion teen, lähti työ etenemään vauhdilla.

Illustrator osuus työstä oli aika suoraviivainen ja tuli ohitettua nopeasti, sillä se oli minulle helppo. Ohjelma oli minulle tuttu ja minulla oli etukäteen selvä mielikuva miten toteutus etenee luonnoksesta valmiiksi. Processing veikin työn toteutuksesta suurimman osan ajasta, sillä se vaati aluksi perehtymistä ja harjoittelua. Mikäli se ei olisi kiin-

nostanut minua niin paljon en olisi varmaan kokenut ehtiväni opetella sitä niin lyhyessä ajassa. Tiesin kuitenkin, että haluaisin jatkossa kokeilla sitä enemmän, joten nyt oli hyvä hetki hankkia perusteet siihen sen lisäksi, että saisin kuvion tekoon vähän lisää näkökulmaa. Minusta oli hyvä idea ottaa Processing Illustratorin rinnalle myös siksi, että ne kaksi vaativat erilaiset lähestymistavat ja näin laajensivat osaamistani kuvioiden teosta. Sain mielestäni käytyä molempien toteutusten vaiheet tarpeeksi perusteellisesti läpi, vaikka toki molemmat edellyttävät pientä ymmärrystä ohjelmoinnista tai vektorigrafiikasta entuudestaan.

Työn loppupuolella päätin vielä kokeilla yhdistää Processing ja Illustrator. Se ei kuulunut alkuperäiseen suunnitelmaani, mutta se tuntui sen verran oleelliselta, että päätin lisätä sen. Siinä ei lopulta mennyt kovin kauan, sillä kyseessä oli hyvin yksinkertainen koodin pätkä. Tein aluksi kokeilun satunnaisella Illustratorissa piirtämälläni kuviolla, ja kun se onnistui niin helposti, päätin vielä kokeilla oikealla kuviollani. Olen tyytyväinen, että päätin kokeilla näiden yhdistämistä. Processingin ja Illustratorin käyttö yhdessä oli niin paljon nopeampaa ja tehokkaampaa, kuin kummankaan käyttäminen yksin. Näiden yhdistämistä voi varioida, esimerkiksi tehdä vain vaikeimmat elementit Illustratorissa ja tuoda useampia pienempiä elementtejä Processingiin, tai sitten tehdä kuten tein tässä ja tuoda kokonaisen elementin.

Alun vaikeuksista huolimatta sain suunniteltua mielestäni mielenkiintoisen kuvion, jonka toteuttaminen onnistui mallikkaasti molemmilla valituilla medioilla. En saanut kaikkia alussa asettamiani tavoitteita toteutettua kuviossa, mutta olen tyytyväinen, että jätin ne matkan varrella. Näin kuvioista tuli omani näköinen mutta ei liian pakotettu. Se ei ollut tylsän yksinkertainen toteutettava, mutta ei myöskään aiheuttanut tarpeettomasti vaikeuksia. Siinä on havainnollistettu teoriaosuudessa käsitetyltäjä asioita, kuten toistamistapoja, elementin luontia ja nimetty mitä Escherin tasoryhmää on käytetty.

Opin työn myötä ohjelmoinnin perusteita Processing-ohjelmointikielellä. Sain myös koottua kasaan kuvioiden teoriaa. Nyt minun on jatkossa helpompi rauhassa suunnitella kuvioita sitten, kun ei ole ennalta asetettuja paineita kuten nyt. Työssä on esitetty lyhyesti tarpeellinen perustieto kuvioiden suunnittelusta, jota saisi muuten etsiä eri lähteistä. Näin siitä on hyötyä myös muille alan opiskelijoille, joita aihe kiinnostaa. Processingin käyttö kuvioiden suunnitteluun voisi myös toimia siltana vaatesuunnittelun ja

graafisen suunnittelun välillä. Toistuvat kuviot mielletään monesti enemmän vaateteollisuuden puolelle, vaikka niiden suunnittelu kuuluu myös graafisen suunnittelun piiriin.

Suorana jatkona tälle työlle voisi suunnitella enemmän kuvioita, joko kaikki hyvin erilaisia keskenään tai sarjan samankaltaisia kuvioita, kuten alun perin suunnittelin ennen työn rajausta. Keskenään hyvin erilaisia kuvioita suunnitellessa Processingin ja Illustratorin erot nousisivat paremmin esille, ja niiden yhdistämistä voisi tutkia monipuolisemmin. Processingin osalta työtä voisi jatkaa vaikka kuinka. Processingin nettisivuilla onkin jo ohjelmoitu eri fraktaaleja, mm. Koching käyrä, sekä Penrosen laatoista yksi laatta setti. Mielestäni myös Escherin tasoryhmiä voisi tarkastella lähemmin, esimerkiksi tutkia, miten sääntöjä saisi ohjelmoitua. Niistä voisi kehittää automaattisia pohjia, mitä sitten voisi täyttää ulkopuolelta tuoduilla tai Processingissa piirretyillä visuaalisilla elementeillä. Hieman samalla idealla, kuin Escher Web Sketch, mutta paljon vapaammin muokattavampana pohjana.

Lähteet

Beyer, Jinny 1999.
Designing tessellations: the secrets of interlocking patterns. Chicago: Contemporary Publishing Group, Inc.

Bunce, Gillian & Phillips, Peter 1993.
Repeat Patterns: a manual for designers, artists and architects. Lontoo: Thames and Hudson Ltd.

Ernst, Bruno 1989.
M.C. Escherin taikapeili. Suom. Hösch, Pirkko. Berliini: TACO.

Frettlöh, D. 2006. Penrose Rhomb. [verkkodokumentti]
<http://tilings.math.uni-bielefeld.de/substitution_rules/penrose_rhomb>
(luettu 6.1.2012)

Grünbaum, Branko & Shephard, G.C. 1987.
Tilings and Patterns. New York: W. H. Freeman and Company.

Heikerö, Markus 2001.
Taiteilijan kuvakirja. Keuruu: Otavan kirjapaino Oy.

Helmberg, Gilbert 2007.
Getting Acquainted with Fractals. Berliini: Walter de Gruyter.

Järvinen, Reijo 2009. Keskuskadusta uusi kävelykatu Helsinging keskustaan. Betoni, 3/2009, 68-78.

Kivelä, Simo 2000.
M niinkuin matematiikka: lukiotason matematiikan tietosanakirja. [PDF-tiedosto]
<http://matta.hut.fi/matta2/isom/pdf2/isom.pdf> (luettu 6.1.2012)

Korkeila, Sampo 2007.
Illustrator CS3 vektorigrafiikka. Helsinki: WSOY.

Michell, George 2007.
The Majesty of Mughal Decoration: The Art and Architecture of Islamic India. Lontoo: Thames and Hudson Ltd.

Hwang, Eric 2012. Penrose Tilings. [verkkodokumentti]
<http://intendo.net/penrose/info_2.html> (luettu 6.1.2012)

Processing.org 2012. About. [verkkodokumentti]
<http://processing.org/about> (luettu 4.1.2012)

Pusa, Unto 1979.
Plastillinen sommittelu. Neljäs tarkistettu painos. Espoo: Otapaino.

Reas, Casey & Fry, Ben 2007.
Processing: A Programming Handbook for Visual Designers and Artists. Cambridge: The MIT Press.

Sanakirja.org 2012. Pattern. [verkkodokumentti]
<http://www.sanakirja.org/search.php?q=pattern&l=3&l2=17> (luettu 24.1.2012)

Suomisanakirja.fi 2012[A]. Kuosi. [verkkodokumentti]
<http://suomisanakirja.fi/kuosi> (luettu 24.1.2012)

Suomisanakirja.fi 2012[B]. Ornamentti. [verkkodokumentti]
<http://suomisanakirja.fi/ornamentti> (luettu 24.1.2012)

Totally Tesselated 2012. Regular Tesselations. [verkkodokumentti]
<http://library.thinkquest.org/16661/simple.of.regular.polygons/regular.1.html>
(luettu 6.1.2012)

Weisstein, Eric 2012. Golden Triangle. [verkkodokumentti]
<http://mathworld.wolfram.com/GoldenTriangle.html> (luettu 6.1.2012)

Kuvalähteet: Silja Karesto

Valmiin kuvion Processing koodi

```
size(600,600);
background(255);
smooth();
for (int y = -54; y < 600; y += 248) {
  for (int x = -30; x < 600; x += 142) {
fill(225,235,90);
noStroke();
beginShape();
vertex(x+162, y+112);
vertex(x+155, y+93);
vertex(x+162, y+74);
vertex(x+172, y+80);
vertex(x+165, y+93);
vertex(x+172, y+106);
endShape(CLOSE);
beginShape();
vertex(x+8, y+20);
vertex(x+28, y+20);
vertex(x+40, y+3);
vertex(x+31, y);
vertex(x+23, y+10);
vertex(x+10, y+10);
endShape(CLOSE);
beginShape();
vertex(x+80, y+42);
vertex(x+100, y+42);
vertex(x+112, y+60);
vertex(x+102, y+62);
vertex(x+94, y+52);
vertex(x+81, y+52);
endShape(CLOSE);
strokeWeight(5);
stroke(90,10,15);
noFill();
beginShape();
vertex(x,y+52);
vertex(x+10,y+10);
```

```
vertex(x+23,y+10);
vertex(x+31,y);
vertex(x+71,y+10);
vertex(x+81,y+52);
vertex(x+94,y+52);
vertex(x+102,y+62);
vertex(x+142,y+52);
vertex(x+172,y+80);
vertex(x+165,y+93);
vertex(x+172,y+106);
vertex(x+142,y+134);
endShape();
line(x+23, y+112, x+11, y+93);
line(x+11, y+93, x+23, y+74);
line(x+8, y+20, x+28, y+20);
line(x+28, y+20, x+40, y+3);
line(x+80, y+42, x+100, y+42);
line(x+100, y+42, x+112, y+60);
vertex(x+70, y+76);
vertex(x+66, y+86);
vertex(x+55, y+84);
endShape(CLOSE);
beginShape();
vertex(x+34, y+120);
vertex(x+55, y+104);
vertex(x+66, y+102);
vertex(x+70, y+112);
vertex(x+66, y+138);
vertex(x+40, y+146);
endShape(CLOSE);
beginShape();
vertex(x+80, y+93);
vertex(x+87, y+84);
vertex(x+112, y+75);
vertex(x+132, y+93);
vertex(x+112, y+111);
vertex(x+87, y+102);
```

```
endShape(CLOSE);
  }
}
for (int y = 70; y < 600; y += 248){
  for (int x = -101; x < 600; x += 142){
    noStroke();
beginShape();
vertex(x+162, y+112);
vertex(x+155, y+93);
vertex(x+162, y+74);
vertex(x+172, y+80);
vertex(x+165, y+93);
vertex(x+172, y+106);
endShape(CLOSE);
beginShape();
vertex(x+8, y+20);
vertex(x+28, y+20);
vertex(x+40, y+3);
vertex(x+31, y);
vertex(x+23, y+10);
vertex(x+10, y+10);
endShape(CLOSE);
beginShape();
vertex(x+80, y+42);
vertex(x+100, y+42);
vertex(x+112, y+60);
vertex(x+102, y+62);
vertex(x+94, y+52);
vertex(x+81, y+52);
endShape(CLOSE);
strokeWeight(5);
stroke(90,10,15);
noFill();
beginShape();
vertex(x,y+52);
vertex(x+10,y+10);
vertex(x+23,y+10);
vertex(x+31,y);
```

```
vertex(x+71,y+10);
vertex(x+81,y+52);
vertex(x+94,y+52);
vertex(x+102,y+62);
vertex(x+142,y+52);
vertex(x+172,y+80);
vertex(x+165,y+93);
vertex(x+172,y+106);
vertex(x+142,y+134);
endShape();
line(x+23, y+112, x+11, y+93);
line(x+11, y+93, x+23, y+74);
line(x+8, y+20, x+28, y+20);
line(x+28, y+20, x+40, y+3);
line(x+80, y+42, x+100, y+42);
line(x+100, y+42, x+112, y+60);
fill(225,235,90);
beginShape();
vertex(x+34, y+66);
vertex(x+40, y+42);
vertex(x+66, y+50);
vertex(x+70, y+76);
vertex(x+66, y+86);
vertex(x+55, y+84);
endShape(CLOSE);
beginShape();
vertex(x+34, y+120);
vertex(x+55, y+104);
vertex(x+66, y+102);
vertex(x+70, y+112);
vertex(x+66, y+138);
vertex(x+40, y+146);
endShape(CLOSE);
beginShape();
vertex(x+80, y+93);
vertex(x+87, y+84);
vertex(x+112, y+75);
vertex(x+132, y+93);
vertex(x+112, y+111);
vertex(x+87, y+102);
endShape(CLOSE);
  }
}
```