|  |
| --- |
|  |

Lyriikkarenki

Arto Pöllänen

Tutkintosuoritus, Tele- ja viestintätekniikan ammattitutkinto

Porvoo International College

29.3.2017

# Johdanto

Tutkintosuoritukseni aiheena on kehittää nettisovellus "Lyriikkarenki", joka suoltaa sanoituksia, jotka rytmisesti, äänneasultaan ja riimitykseltään sopivat annettuun malliin. Malli on joko oikea sanoitus tai käyttäjän kirjoittama vapaavalintainen teksti. Sovellus selvittää kappaleen rytmityksen, tavujen pituudet ja mahdollisen riimien käytön. Sikäli kun mallisanoituksen sanat löytyvät tietokannasta, myös niiden sanaluokat ja taivutusmuodot selviävät. Sovellus tekee sitten sanoituksen, joka teknisesti istuu malliin, jolloin uusi sanoitus on mahdollista laulaa samalla sävellyksellä ja sovituksella kuin alkuperäinenkin. Sisältö on tietysti enimmäkseen non-senseä, mutta sovellus synnyttää paljon uusia yllättäviä kielikuvia.

Aihe kiinnostaa minua, koska harrastan sanoittamista ja olen myös tutkinut sanastojen ja tekstikokoelmien ohjelmallista käsittelyä. Minulla on käytettävissäni laaja sanasto, jossa suomenkielen Perussanakirjan sanasto on taivutettuna kaikkiin taivutusmuotoihinsa. Tämän sanaston laajuus on yli 20 000 000 sanaa. Olen aiemmin toteuttanut nettisovelluksen "Riimirenki", joka pohjautuu tähän samaan sanastoon (http://elvisry.fi/riimirenki/riimirenkipro.php).

Netissä on paljon runogeneraattoreita:

*http://www.yostaja.fi/runorobotti*

*http://www.runokone.com/runokone/*

*http://www.ioni.org/runokone/index.php*

*http://www.tarinakaruselli.fi/runogeneraattori/*

*http://yle.fi/ylex/uutiset/somessa\_supersuosion\_saanut\_oletrunofi-sivusto\_on\_runoillut\_jo\_miljoonalle\_kayttajalle/3-7579300*

Niissä kysytään yleensä käyttäjältä sanoja, joita halutaan runoon ja generaattori muodostaa runon arpoen sanoja valmiiseen malliin. Lyriikkarengin lähtökohta on erilainen, koska siinä yritetään kopioida mallisanoituksen rakenne mahdollisimman tarkasti.

# Projektin toteutunut aikataulu

vko 8: perehtyminen "kilpailijoihin", nettisivun perustaminen

vko 9: toiminnallisuuden suunnittelu, tietokantarakenteen määrittely

vko 10: sanastojen lataaminen tietokannan tauluihin, käyttöliittymän suunnittelu

vko 11-12: toteutus ja testaus

vko 13: dokumentointi

# Tieto- ja viestintäteknisessä toimintaympäristössä toimiminen

Sovelluksen toteutus ja julkaiseminen osoittanevat että pystyn toimimaan tieto- ja viestintäteknisessä toimintaympäristössä. Point Collegessa opiskelun lisäksi minulla on karttunut kokemusta monenlaisista toimintaympäristöistä työskennellessäni IT-alan tuotekehitys- ja ylläpitotehtävissä kahdentoista työnantajan palveluksessa 32 vuoden aikana Suomessa, USA:ssa ja Ranskassa.

# Toiminnallisuus

## Mallisanoitus

Sovelluksessa on teksti-ikkuna, johon käyttäjä voi kirjoittaa mallisanoituksen. Mallisanoituksen on oltava tavutettu, koska kappaleen rytmityksessä on oleellista se, kuinka tekstin sanat on jaettu tavuihin. Sovelluksessa on painike "Tavuta mallisanoitus", joka poistaa mahdolliset vanhat tavuviivat ja tavuttaa mallisanoituksen suomenkielen tavutussääntöjen mukaan. Tavutus on syytä silti vielä tarkistaa, koska kappaleen rytmitys ei ehkä olekaan vielä haluttu.

Sovelluksessa on alasvetovalikko "Valitse mallisanoitus". Sitä käyttäen voi valita mallisanoituksen oikeiden sanoitusten listalta. Nämä sanoitukset on valmiiksi tavutettu ja tavutuksen istuvuus kappaleen rytmiin on tarkistettu.

## Tavutus

Tavutusalgoritmi on ainoa osa sovellusta, jota en ole koodannut alusta lähtien itse, vaan olen ottanut mallia Topi Laamasen harjoitustyöstä (*http://topin.fi/lahdekoodit/C/tavutus.c*). Topin koodi oli kirjoitettu C-kielellä ja minä muokkasin siitä PHP-kielisen koodin.

Suomen kielen tavutussäännöt:

1. Konsonanttisääntö: Jos tavuun kuuluvaa vokaalia seuraa yksi tai useampia konsonantteja, joita vielä seuraa vokaali, tavuraja sijoittuu välittömästi ennen viimeistä konsonanttia.

2. Vokaalisääntö: Jos tavun ensimmäistä vokaalia seuraa toinen vokaali, niiden väliin tulee tavuraja, ellei

a) edellinen vokaali ole sama kuin jälkimmäinen (pitkä vokaali).

b) jälkimmäinen vokaali ole i (i:hin loppuva diftongi).

c) kysymyksessä ole jokin vokaalipareista au, eu, ie, iu, ou, uo, yö, äy, öy, ey tai iy (muu diftongi).

3. Diftongisääntö: Jos tavun kuuluvaa diftongia tai pitkää vokaalia seuraa vokaali, tähän väliin tulee aina tavuraja.

4. Poikkeussääntö: Yhdyssanat jaetaan tavuihin sanojen välistä, myös siinä tapauksessa että sana on yhdyssana vain alkuperäiskielessä.

## Tulossanoitus

Ensin valitaan valintaruuduilla, mitä sanastoja halutaan käyttää laajimman sanaston lisäksi. (K18, biiseistä luettu, yleisimmät sanat). Mallisanoitusta vastaavan tulossanoituksen generointi käynnistyy painikkeella "Muodosta tulossanoitus":

* Jos mallisanoituksessa ei ole tavuviivoja, se tavutetaan ensin automaattisesti.
* Jokaisesta mallisanoituksen sanasta lasketaan tavujen pituudet.
* Jokainen mallisanoituksen sana etsitään laajasta sanastosta ja jos se löytyy, otetaan talteen sanaluokka, taivutusmuoto ja yksikkö/monikko -tieto.
* Jokaiselle mallisanoituksen sanalle pyritään löytämään mallisanoituksesta "riimipari" ja mikä "riimityyppi" parille sopii. (Riimi on tässä tavallista joustavampi käsite),
* Riimityyppejä ovat tässä IDENTTINEN RIIMI, PUHDAS RIIMI, RAPVOKAALIRIIMI, KONSONANTTIRIIMI, ALKURIIMI, SAMA SANA ja MUUTTUMATON SANA. Riimityypillä MUUTTUMATON SANA ei ole riimiparia.
* Riimityypit etsitään tietyssä järjestyksessä. Riimiparin riimityyppi merkitään aina parin jälkimmäiseen sanaan. Jos sanalle on jo löydetty riimityyppi, se ”lukitaan" eli sen tutkiminen lopetetaan. Etsimisjärjestys eli riimityyppien prioriteettijärjestys:
* Etsitään kaikki identtiset sanat ja muodostetaan niistä SAMA SANA -riimiparit.
* Verrataan joka rivin viimeistä sanaa neljän edellisen rivin viimeisiin sanoihin (järjestys: alhaalta ylöspäin) ja merkitään löytynyt PUHDAS RIIMI ja myös rimmaavien tavujen lukumäärä.
* Verrataan joka rivin viimeistä sanaa neljän edellisen rivin viimeisiin sanoihin (järjestys: alhaalta ylöspäin) ja merkitään löytynyt IDENTTINEN RIIMI ja myös rimmaavien tavujen lukumäärä.
* Verrataan jokaista sanaa kaikkiin saman rivin edeltäviin sanoihin (järjestys: oikealta vasemmalle) ja merkitään löytynyt PUHDAS RIIMI ja myös rimmaavien tavujen lukumäärä,
* Verrataan jokaista sanaa kaikkiin saman rivin edeltäviin sanoihin (järjestys: oikealta vasemmalle) ja merkitään löytynyt IDENTTINEN RIIMI ja myös rimmaavien tavujen lukumäärä.
* Verrataan jokaista sanaa neljään edelliseen saman rivin sanaan (järjestys: oikealta vasemmalle) ja merkitään löytynyt RAPVOKAALIRIIMI.
* Verrataan jokaista sanaa neljään edelliseen saman rivin sanaan (järjestys: oikealta vasemmalle) ja merkitään löytynyt KONSONANTTIRIIMI.
* Verrataan jokaista sanaa neljään edelliseen saman rivin sanaan (järjestys: oikealta vasemmalle) ja merkitään löytynyt ALKURIIMI ja myös sanojen alun identtisten kirjainten lukumäärä.
* Jos sana on joku näistä: MUN, SUN, ME, TE, MÄ, SÄ, EI, EN, ET, JA, JO, JOS, KAI, KUN, MYÖS, TAI, ON, SE, SEN, NYT, merkitään riimityypiksi MUUTTUMATON SANA.
* Jos sana alkaa jollakin näistä aluista: MI-NÄ, SI-NÄ, MI-NU, SI-NU, merkitään riimityypiksi MUUTTUMATON SANA.
* Tämän jälkeen aletaan muodostaa tulossanoitusta. Käydään mallisanoitus läpi sana sanalta vasemmalta oikealle ja ylhäältä alas. Aina kun sana on saatu muodostettua, niin päivitetään Tulossanoitus-ikkunaan koko siihen asti muodostettu tulossanoitus, samoin päivitetään etenemispalkki ja prosenttiluku Mallisanoitus- ja Tulossanoitusikkunoiden yläpuolella.
* Jos sanan riimityyppi on MUUTTUMATON SANA, kopioidaan tulossana suoraan mallisanoituksesta.
* Jos sanan riimityyppi on SAMA SANA, kopioidaan tulossana riimiparin toisesta päästä.
* Muilla riimityypeillä ja riimityypittömillä sanoilla sopiva *tulossana haetaan sanastoista.*
* Jos sanan riimityyppi on PUHDAS RIIMI tai IDENTTINEN RIIMI ja tulosanaa ei löytynyt, kokeillaan vaihtaa riimiparin toisen pään sanaa ja yritetään sitten uudestaan. Toistetaan tätä korkeintaan 10 kertaa. Jos tämäkään ei auttanut ja riimitavuja on kolme tai enemmän, vähennetään niitä kahdella ja yritetään vielä korkeintaan 10 kertaa.
* *tulossana haetaan sanastoista*
* Sopivia sanoja etsitään niistä sanastoista, mitä käyttäjä on valinnut järjestyksessä ylhäältä alas (k18, biiseistä, yleiset) ja jos niistäkään ei löytynyt, etsitään laajasta sanastosta. Hakuehtoina ovat tavumäärä, tavujen pituudet, sanaan luokka/taivutus/luku (jos nämä on pystytty selvittämään mallisanoituksen sanasta) ja riimiehto. Haettavien sanojen maksimimäärä on 500. (Näin haut ei hidastu kohtuuttomasti.)
* Jos tulossanaa ei löydy laajastakaan sanastosta, hakuehtoja helpotetaan asteittain, tavumäärän pysyessä kuitenkin aina hakuehtona.
* Laajalla sanastolla riittää, että haun tuloksena saadaan vähintään yksi sana. Muilla sanastoilla osumia pitää olla ainakin neljä (tällä pyritään estämään samojen sanojen toistuminen tulossanoituksissa).
* Haun tuloksia aletaan lukea satunnaisessa (arvotussa) järjestyksessä. Tarkastetaan, onko PUHDAS RIIMI varmasti puhdas ja IDENTTINEN RIIMI varmasti identtinen, ja ettei sana ole sama kuin mallisanoituksessa ja muutama muu tarkistus. Ensimmäinen ehdot täyttävä sana hyväksytään tulossanaksi.

# Tietokanta

Tietokantana käytetään MySQL-kantaa.

## Sanastotaulut

laaja sanasto:

lr\_sanasto1 // 421 1-tavuista sanaa

lr\_sanasto2 // 46 228 2-tavuista sanaa

lr\_sanasto3 // 532 137 3-tavuista sanaa

lr\_sanasto4 // 2 120 188 4-tavuista sanaa

lr\_sanasto5 // 4 543 220 5-tavuista sanaa

lr\_sanasto6 // 6 073 280 6-tavuiset sanat

lr\_sanastoX // 7 535 355 7-tavuista ja sitä pitempää sanaa

muut sanastot:

lr\_sanasto\_biiseistä // 102800 1-16 -tavuista sanaa jotka on koottu 3944 sanoituksesta

lr\_sanasto\_yleiset // 5545 1-4 -tavuista sana suomenkielen yleisimpien sanojen tilastosta

lr\_sanasto\_k18 // 18300 2..4 -tavuista alatyylin sanaa

Laaja sanasto (lr\_sanasto1..6 ja lr\_sanastoX) perustuu suomenkielen Perussanakirjan sanastoon jonka joku harrastaja on koneellisesti taivuttanut kaikkiin taivutusmuotoihinsa. Kun Perussanakirjassa on ehkä 100 000 sanaa, on taivutettuja sanoja yli 20 000 000. Näistä sanoista tiedetään myös sanan yksikkö/monikko, sanaluokka ja taivtusmuoto. Muissa sanastoissa (lr\_sanasto\_biiseistä, lr\_sanasto\_yleiset ja lr\_sanasto\_k18) ovat nämä tiedot mukana vain silloin, kun sama sana löytyy myös laajasta sanastosta.

Sanastotauluissa on seuraavat kentät:

- id // (INT) uniikki tunniste

- tavuja // (INT) tavumäärä (ei tauluissa lr\_sanasto1..6)

- tavunpituudet // (INT) sanan tavujen pituudet koodattuna kokonaislukuun

- luokka // (VARCHAR(5)) merkkijono johon on koodattu sanaluokka ja taivutusmuoto ja yksikkö/monikko

- crc // (INT) tavuja, tavunpituudet ja luokka -kenttien sisällöistä laskettu crc32-typpinen tarkistussumma. Tämä nopeuttaa hakuja huomattavasti, koska sovellus hakee hyvin usein sanoja noiden kenttien tietyillä arvoilla.

- sana // VARCHAR(50) sana tavuviivoineen

- tavutonsana // VARCHAR(50) sana ilman tavuviivoja

- takaperin // VARCHAR(50) sana tavuviivoineen takaperin. Tämä nopeuttaa riimien hakuja, joissa haetaan nimenomaan tiettyä sanan loppuosaa.

Sanastotauluille on määritelty seuraavat indeksit nopeuttamaan useimmin esiintyviä hakuja. Indeksin nimestä voi päätellä, mitä kenttiä se sisältää ja missä järjestyksessä.

- PRIMARY

- id\_UNIQUE

- crc

- crc\_takaperin

- crc\_tavutonsana

- tavuja\_tavunpituudet\_takaperin

- tavuja\_tavunpituudet\_tavutonsana

- tavuja\_takaperin

- tavuja\_tavutonsana

Huom! Tauluille lr\_sanasto1..6 on muuten samat indeksit, mutta indekseissä ei ole tavuja-kenttää:

- PRIMARY

- id\_UNIQUE

- crc

- crc\_takaperin

- crc\_tavutonsana

- tavunpituudet\_takaperin

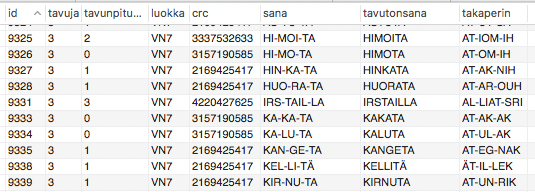
- tavunpituudet\_tavutonsana

- takaperin

- tavutonsana

Esimerkki taulun lr\_sanasto3 -sisällöstä:



Esimerkki taulun lr\_sanasto\_k18 sisällöstä:  


Muut taulut

Sanastotaulujen lisäksi kannassa on taulu lr\_mallisanoitukset, joka pitää sisällään sanoitukset, joista käyttäjä voi halutessaan valita mallisanoituksen. Taulun kentät:

- id // (INT) uniikki tunniste

- biisinnimi // VARCHAR(50) mallisanoituksen nimi

- esittaja // VARCHAR(50) mallisanoituksen tunnetuin esittäjä

- genre // VARCHAR(50) musiikkityypin nimi, ei vielä käytössä

- sanoitus // LONGTEXT sanoitus tavutettuna ja rivitettynä

Esimerkki taulun lr\_mallisanoitukset sisällöstä:  


# Toteutus

Koska halusin responsiivisen ja tyylikkään nettipalvelun, ei kannattanut koodata sitä pelkällä HTML:llä vaan käytin mieluummin valmista kehikkoa. Ensimmäisenä tuli mieleeni Wordpress, mutta se tuntui kuitenkin aivan liian raskaalta näin yksinkertaiseen tarpeeseen. Lopulta päädyin w3schools.com:min kehittämään W3.CSS :ään. Se on kevyt, ilmainen, helppo oppia, käyttää vain standardi-CSS:ää, mutta sillä saa rakennettua tyylikkäät ja responsiiviset sivut.

Koodauskieleksi valitsin PHP:n : siinä on hyvä MySQL-API, ja se soveltuu erinomaisesti tekstimuotoisen datan käsittelyyn. Tämän lisäksi tarvittiin kaksi javascriptillä kirjoitettua funktiota. Kirjoitin lähdekoodit Visual Studio Code -työkalulla.

Sanaris Oy luovutti ystävällisesti käyttööni levytilaa palvelimeltaan sekä MySQL-tietokannastaan. Osoitteessa *http://mittatilauslaulut.fi/lr* on neljä tiedostoa:

index.html // dummy-tiedosto, joka estää hakemiston tutkimisen ulkupuolisilta

lyriikkarenki.php // sovelluksen W3.CSS-käyttöliittymä, enimmäkseen html-koodia. Php:llä toteutettu toiminnallisuus on lr\_malli.php:ssä ja lr\_tulos.php:ssä, jotka liitetään lyriikkarenki.php:hen include-komennoilla.

lr\_malli.php // mallisanoituksen käsittelyyn tarvittavat php-funktiot, lisäksi yleiskäyttöisten funktioiden kirjasto

lr\_tulos.php // tulossanoituksen muodostamiseen tarvittavat php-funktiot

Sovellus on kokeiltavissa julkisessa netissä osoitteessa *http://mittatilauslaulut.fi/lr/lyriikkarenki.php*

Php-funktiot

*lr\_malli.php :*

function tulostaMallisanoituslista() // luo käyttöliittymään ”Valitse mallisanoitus” -alasvetovalikon

function tavutaMallisanoitus($mallisanoitus) // palauttaa annetun sanoituksen tavutettuna

function tavutaSana($sana, &$tavutettu) // tavuttaa sanan suomenkielen tavutussääntöjen mukaan

// yleiskäyttöisiä merkkijonofunktioita:

function onkoKirjainTaiTavuviiva($m)

function onkoKirjain($m)

function onkoIsoKirjain($m)

function isoiksiSkandeineen($input)

function pieniksiSkandeineen($input)

function sisaltaakoNaitaMerkkeja($str, $merkit)

function laskeTavunpituudet($sana)

function skanditTextAreasta($text)

function skanditTextAreaan($text)

function onkoVokaali($k)

function onkoPitkaTavu($tavu)

function onkoVokaalipari($k1, $k2)

function startsWith($haystack,$needle)

function endsWith($string,$test)

function debug($msg) // VAIN TESTAUSTA VARTEN

function debugR($msg) // VAIN TESTAUSTA VARTEN

*lr\_tulos.php*

function muodostaTulossanoitus($mallisanoitus) // tekee varsinaisen työn, eli muodostaa tulossanoituksen mallisanoituksen perusteella

function tahanastinenTulos($rr, $ss) // rakentaa tulossanoituksen (myös keskeneräisen) biisi-taulukon perusteella

function analysoiSana($sana, $edeltavatMerkit, $r, $s) // etsi annetun sanan tiedot tietokannasta

function analysoiRiimitys() // selvittää mallisanoituksen riimitykset

function rapVokaalit($sana) // palauttaa sanan vokaalit, vaihtaen pitkät vokaalit lyhyiksi

function konsonantit($sana) // palauttaa sanan konsonantit

function laskePuhtaatLoppuriimitavut($sana1, $sana2) // laskee montako rimmaavaa loppuriimitavua sanoilla on

function laskeIdenttisetLoppuriimitavut($sana1, $sana2) // laskee montako yhteistä tavua sanojen lopuissa on

function rimmaakoTavut($tavu1, $tavu2) // laskee muodostavatko tavut puhtaan riimiparin

function haePuhdasTaiIdenttinenRiimi($r, $s) // yrittää löytää sanalle riimiparin, tarvittaessa vaihtaen sanaa

function haeSanastoista($r, $s, $riimi) // yrittää löytää sanalle riimiparin kaikista valituista sanastoista

function haeYhdestaSanastosta($r, $s, $riimi, $sanasto) // yrittää löytää sanalle riimiparin yhdestä sanastosta

function teePuhtaanRiiminHakusana($riimi, $riimitavut) // tekee puhtaan riimin hakusanan MySQL:n select-lausetta varten

function teeIdenttisenRiiminHakusana($riimi, $riimitavut) // tekee identtisen riimin hakusanan MySQL:n select-lausetta varten

function arvoSana($sql, $sanasto, $p, $riiminToinenPaa, $palautaJokuSana) // hakee sanoja sql-lauseen perusteella ja arpoo niistä yhden palautettavan sanan

function haeSanaKannasta($tavujenLkm, $id, $sanasto) // VAIN TESTAUSTA VARTEN

function mysqlQuery($sql) // läpikutsu, sama kuin mysql\_query($sql,$yhteys)

Javascript-funktiot

*lr\_tulos.php*

function paivita\_tulossanoitus(eteneminen, sanojenLkm, tulossanoitus) // päivittää tulossanoituksen käyttöliittymään, myös keskeneräisen

function naytaOhje() // näyttää käyttöohjeen käyttöliittymässä

*lyriikkarenki.php*

function mallisanoitusValittu() // vaihtaa valitun mallisanoituksen käyttöliittymään

Luokkamäärittelyt

*lr\_tulos.php*

class SanaStruct { // biisi-taulukon alkio: mallisanoituksen/tulossanoituksen sana

public $mallisana; // textAreasta

public $mallisanaIsoilla; // isot kirjaimet, skandit kuten kannassa

public $tavujenLkm; // tavujen lkm

public $edeltavatMerkit; // merkit ennen sanaa

public $luokka; // kannasta haettu luokka, jos ei löytynyt, tyhjä stringi

public $tavunpituudet; // tavujenpituuskoodi

public $riimityyppi; // EI\_RIIMIA, IDENTTINEN\_RIIMI, PUHDAS\_RIIMI jne.

public $riimitavut; // jos riimi, sen pituus tavuina, muuten 0

public $riimisarake; // jos riimi, monesko rivin sana se on (0..), muuten -1

public $riimirivi; // jos riimi, monesko rivin sana se on (0..), muuten -1

public $samaaAlkua; // alkuriimeillä, montako samaa alkukirjainta

public $rapVokaalit; // mallisanaIsoilla -sanan vokaalit, tuplat poistettuna

public $konsonantit; // mallisanaIsoilla -sanan konsonantit

public $tulossanaIsoilla; // kannasta haettu tulossana isoilla

public $tulossana; // kannasta haettu tulossana lopullisella casella

}

Pohdinta

Olin yhteydessä Matti Huhtaan (räppäri Sere/Seremoniamestari). Hän piti tätä hauskan kuuloisena projektina ja arveli, että hänestä voisi olla apua sovelluksen "markkinoinnissa".

Kävin esittelemässä Lyriikkarenkiä myös laulaja- ja lauluntekijä Heikki Salolle (mm. Miljoonasade). Tulimme Heikin kanssa siihen tulokseen, että tällaisenaan tätä ei ole syytä vielä esitellä suurelle yleisölle: tulossanoitukset ovat liian non-senseä, jotta niistä olisi hyötyä sanoittajille edes ideoiden antajana. Tulossanoituksia pitäisi tiukemmin ohjata tiettyihin lauserakenteisiin ja käyttämään haluttuun teemaan sopivaa sanastoa. Tulossanoitus voisi sisältää myös enemmän mallisanoituksen sanastoa. Yksi mahdollinen kehityssuunta olisi tehdä tämän pohjalta jonkinlainen karaokepeli.

Uutena ajatuksena on kehittää Lyriikkarengistä "sanoittajan Autotune": sovellukselle annetaan sanoitus sekä jollakin menetelmällä rytmitys, johon sanoituksen olisi istuttava. Sovellus paljastaisi virheet tai jopa yrittäisi muokata sanoituksen rytmiin sopivaksi.