Aalto-yliopisto
Perustieteiden korkeakoulu
Tietotekniikan koulutusohjelma

# WebGL HTML5-pelinkehittäjän näkökulmasta

**Kandidaatintyö**

**14. helmikuuta 2014**

**Atte Isopuro**

Aalto-yliopisto
Perustieteiden korkeakoulu
Tietotekniikan koulutusohjelma

KANDIDAATINTYÖN
TIIVISTELMÄ

| | |
|---|---|
| **Tekijä:** | Atte Isopuro |
| **Työn nimi:** | WebGL HTML5-pelinkehittäjän näkökulmasta |
| **Päiväys:** | 14. helmikuuta 2014 |
| **Sivumäärä:** | TODO: Kirjoita tähän lopuksi oikea määrä, tässä esimerkissä 23 |
| **Pääaine:** | Ohjelmistotekniikka |
| **Koodi:** | T3001 |
| **Vastuuopettaja:** | Ma professori Tomi Janhunen |
| **Työn ohjaaja(t):** | Professori Jukka Nurminen (Tietoliikenneohjelmistot) |
| TODO: tiivistelmä | |
| **Avainsanat:** | webgl, html5, peli, pelinkehitys, suorituskyky, ongelmia, edut |
| **Kieli:** | Suomi |

Aalto-universitetet
Högskolan för teknikvetenskaper
Examensprogrammet för datateknik

SAMMANDRAG AV
KANDIDATARBETET

| | |
|---|---|
| **Utfört av:** | Atte Isopuro |
| **Arbetets namn:** | WebGL HTML5-pelinkehittäjän näkökulmasta |
| **Datum:** | 14. helmikuuta 2014 |
| **Sidoantal:** | TODO: Kirjoita tähän lopuksi oikea määrä, tässä esimerkissä 23 |
| **Huvudämne:** | Ohjelmistotekniikka |
| **Kod:** | T3001 |
| **Övervakare:** | Ma professori Tomi Janhunen |
| **Handledare:** | Professori Jukka Nurminen (Tietoliikenneohjelmistot) |
| Ett abstrakt hit | |
| **Nyckelord:** | webgl, html5, peli, pelinkehitys, suorituskyky, ongelmia, edut |
| **Språk:** | Svenska |

# Sisältö

# 1 Johdanto

Tämä kandidaatintyö käsittelee HTML5:n WebGL-ohjelmointirajapintaa pelinkehittäjän näkökulmasta.

HTML5-spesifikaatio on mahdollistanut kaikenlaisen median esittämisen verkossa ilman että käyttäjän tarvitsee asentaa erillisiä lisäosia. Yksi näistä uusista rajapinnoista on WebGL jonka kautta verkkosivun on mahdollista käyttää tietokoneen näytönohjainta piirtämiseen.

Erityisesti monimutkaista ja tarkkaa interaktiota vaativat pelit tarvitsevat nopeaa grafiikan piirtämistä ruudulle. Alustasta riippumaton grafiikkaohjelmointi on erittäin kiinnostavaa tästä näkökulmasta.

Kirjallisuudessa on käsitelty WebGL:n käyttömahdollisuuksia erinäisillä aloilla. Useimmat tällaiset käsittelyt ovat kuitenkin yksittäisiä sovelluksia: yleistä arviota WebGL:n eduista ja haitoista pelinkehityksessä ei ole tehty.

Tämän kandidaatintyön tavoitteena on kerätä pelinkehittäjän kannalta hyödyllisiä havaintoja WebGL:stä. Työssä pyritään tunnistamaan sellaiset WebGL:n ominaispiirteet jotka pelinkehittäjän tulisi ottaa huomioon. Tämä työ ei tule käsittelemään muita selaimen grafiikka-esitykseen liittyviä teknologioita, kuten Canvas 2D context tai Flash. Teknologioita verrataan WebGL:ään, mutta niiden erityispiirteitä ei tutkita tässä työssä tarkemmin. Työssä ei myöskään pyritä toteamaan WebGL:n paremmuudesta mitään konkreettista. Löytöjen perusteella peinkehittäjän on yhä itse päätettävä mikä teknologia soveltuu hänen projektiinsa parhaiten.

Työ toteutetaan kirjallisuuskatsauksena. Aineisto koostuu tieteellisistä artikkeleista ja kirjoista jotka käsittelevät WebGL:ää. Erityisesti viitataan teksteihin joissa on konkreettisten toteutusten yhteydessä havaittu WebGL:n ominaispiirteitä.

Ensiksi alustetaan aiheen taustaa: esitellään HTML5 ja grafiikan piirtäminen. Seuraavaksi selvitetään tarkemmin pelinkehittäjän tarpeet sekä WebGL-spesifikaation tarjoamat edut. Tämän jälkeen eritellään tutkimusmenetelmät: lähteiden rajauskriteerit selvitetään ja kerrotaan kuinka ne on määrätty luotettaviksi. Lopuksi aineistosta kerätty tieto kootaan eheäksi kokonaisuudeksi, josta viimein tullaan lopullisiin johtopäätöksiin.

# 2 Taustaa

Takaisin sisällysluetteloon

## 2.1 HTML5

HTML5 on uusin versio HyperText Markup Language-kielestä. Yksi HTML5:n tuomia uudistuksia on erinäisten media-formaattien sisäistäminen itse spesifikaatioon erinäisten rajapintojen kautta[7]. Näin ollen sisällöntuottajien ja käyttäjien ei tarvitse huolehtia kolmansien osapuolten liitännäisistä kuten Flash. Tällöin sisällöntuottaja voi olla varma siitä että kaikke ne joilla on spesifikaatiota noudattava selain pääsevät käsiksi hänen tuottamaan sisältöön. Täten alustojen erilaisuuksien ei pitäisi ainakaan teoriassa vaikuttaa sovellukseen, jolloin sisällöntuottaja voi keskittyä luomaan sisältöä ainoastaan yhdelle alustalle.

## 2.2 WebGL

WebGL on JavaScript-rajapinta HTML5:n canvas-elementtiä varten. WebGL perustuu pitkälti OpenGL ES:ään (OpenGL Embedded Systems)[4]. Rajapinta sallii näytönohjaimen käytön JavaScript-koodista ilman että sovellusohjelmoijan tarvitsee tietää tarkalleen mistä laittesta on kyse. Näytönohjaimen käyttö mahdollistaa siis raskaan laskennan siirtämisen pois suorittimesta: fysiikan mallinnus ja monimutkaisen grafiikan piirtäminen ovat tällöin mahdollisia ilman että itse käyttöjärjestelmä hidastuu.

## 2.3 Pelinkehittäjän tarpeet

Erityisesti nopeita reaktioita vaativien pelien pelattavuus kärsii huomattavasti liian pienillä ruudun päivitysnopeuksilla: ero pelaajan pelaamiskyvyssä korkeiden ja matalien piirtotaajuuksien välillä on huomattava[1]. Näin ollen pelin sunnittelijan kannattaa rajata pelinsä sisältö sellaiseksi että se pystytään esittämään tarvittavan sujuvassa muodossa. Tällöin joidenkin pelityyppien näyttävyys saattaa rajautua huomattavasti jos ne ovat ollenkaan toteutettavissa. Jos WebGL on toimiva työkalu, se lisäisi pelinkehittäjien ilmaisuvoimaa. Paljon laskentatehoa vaativat pelityypit ovat tähän mennessä olleet pakostakin natiivisovelluksia: jos WebGL toimii hyvin, se mahdollistaisi helpomman ja halvemman kehitystyön tällaisille peleille.

# 3 Tutkimusmenetelmät

Takaisin sisällysluetteloon

Tutkimusaineiston luotettavuus varmistetaan tarkistamalla onko aineisto Scopus-tietokannassa.

# 4 Tutkimusmateriaali

Takaisin sisällysluetteloon

## 4.1 Suorituskyky

Selkein WebGL:n tarjoama etu on suorituskyky: mahdollisuus suorittaa laskentaa näytönohjaimessa suorittimen sijasta on huomattavasti nopeampaa: Canvas 2D Context ja Flash ovat vertailussa selvästi hitaampia[5]. WebGL pärjää myös yllättävän hyvin natiivia toteutusta vastaan. C++-pohjainen yksinkertainen OpenGL-sovellus on WebGL:ää hitaampi: natiivi toteutus täytyy optimoida ennen kuin se on tehokkaampi kuin WebGL-toteutus[5]. Ero ei suurimmaksi osaksi kuitenkaan johdu WebGL:n ja natiivin OpenGL:n välisistä eroista. Koska WebGL:ää kutsutaan verkkosivun koodista, selaimen on suoritettava JavaScript-koodia jokaista piirtoa varten. JavaScript-koodi on muutamia erikoistapauksia lukuunottamatta paljon hitaampaa kuin C++-koodi[6]. Täten suurin osa WebGL:n piirtämisajasta kuluu JavaScriptin kääntämiseen ja ajamiseen[5].

Pelinkehittäjän kannattaa siis ottaa huomioon että WebGL on huomattavasti hitaampi kuin optimoitu natiivi sovellus, mikä oli odotettavissa. Huomionarvoisempaa on kuitenkin että WebGL ei itse ole pääsyyllinen kyseiseen eroon: WebGL-koodin optimoiminen ei merkittävästi vaikuttanut piirtonopeuteen[5]. On huomionarvoista että WebGL-grafiikan optimoimisessa ei välttämättä kannata keskittyä itse WebGL-koodiin: JavaScript-koodin optimoiminen voi tuottaa paljon suuremmat hyödyt. Näin ollen JavaScript-kokemus voi kompensoida puutteita kehittäjän WebGL-kokemuksessa, mikä saattaa huomattavasti helpottaa teknologian käyttöönottoa pelinkehittäjien keskuudessa.

## 4.2 Kirjaston kattavuus

Pelinkehittäjän on aina hyvä tietää mihin valittu teknologia pystyy ja mitä puutteita sillä on. Näin ollen on aiheellista katsastaa WebGL:n funktiokirjastoa, jossa on muutamia puutteita[2]:

WebGL on matalan tason kirjasto joka oletusarvoisesti toimii sekä tietokoneilla että mobiililaitteilla. Tästä syystä siitä saattaa puuttua toiminallisuutta joka löytyy oletusarvona suuremmista kirjastoista. Yksi tärkeä ominaisuus nykyaikaisissa grafiikkakirjastoissa on asynkroninen lataaminen. Kuvaa piirrettäessä tietokoneen on haettava muistista erinäistä tietoa: tekstuureita, malleja ym. Jos tällaiset tiedot pitää ladata yksi kerrallaan ohjelma saattaa lakata reagoimasta käyttäjään kunnes kaikki tarvittava tieto on ladattu. Asynkroninen lataaminen sallii eri osien lataamisen samanaikaisesti, jolloin suoritin voi samalla reagoida käyttäjään. JavaScript salli tällaisen asynkronisuuden ainoastaan Web Workers

rajapinnan kautta[8], joten ominaisuutta ei löydy oletuksena WebGL:ää käyttäessä.

Yksi HTML5:n hyödyllisistä ominaisuuksista ovat elementti-kohtaiset rajapinnat. Esimerkiksi

`<img>`

elemetin JavaScript-rajapinnasta löytyy

`onload`

-funktio, joka sallii kehittäjän määritellä mitä tehdään kun kyseisen elementin määrittelemä kuva on ladattu valmiiksi. JavaScriptissä tai WebGL:ssä ei ole tällaista toiminnallisuutta 3D-objekteille: HTML5 ei määrittele tällaisille objekteille erityisiä elementtejä.

Tärkeä asia pitää mielessä WebGL:ssä on että se perustuu OpenGL ES:ään, joka vuorostaan sisältää vain osan täysimittaisen OpenGL-kirjaston toiminnallisuudesta (kuten OpenGL 3.0)[3]. Näin ollen kehittäjä joka on kokenut OpenGL-koodaaja ei välttämättä ole niin tehokas kuin voisi toivoa: WebGL:n rajallisuus OpenGL:ään verrattuna saattaa vaatia sopeutumista.

Huomattavin puute WebGL:ssä muihin grafiikkakirjastoihin verrattuna on lineaarialgebran puuttuminen. 3D-grafiikan piirtämisessä käytetään huomattavia määriä erilaista matriisilaskentaa. Näitä funktioita ei löydy WebGL:stä, joten kehittäjän on joko toteutettava ne itse tai käytettävä kolmannen osapuolen kehittämää kirjastoa.

## 4.3 Alustariippumattomuus

# 5 Arviointi

Takaisin sisällysluetteloon

## 5.1 Riskit

## 5.2 Hyödyt

# 6 Johtopäätökset

Takaisin sisällysluetteloon

# Lähteet


[1] Mark Claypool, Kajal Claypool ja Feissal Damaa. The effects of frame rate and resolution on users playing first person shooter games. *Electronic Imaging 2006*, sivut 607101–607101–11. International Society for Optics and Photonics, 2006. Saatavissa `http://web.cs.wpi.edu/~claypool/papers/fr-rez/paper.pdf`. Viitattu 15.02.2014. DOI: 10.1117/12.648609.

[2] Marco Di Benedetto, Federico Ponchio, Fabio Ganovelli ja Roberto Scopigno. Spidergl: a javascript 3d graphics library for next-generation www. *Proceedings of the 15th International Conference on Web 3D Technology*, sivut 165–174. ACM, 2010. Saatavissa ACM Digital Library, DOI: 10.1145/1836049.1836075. Viitattu 18.02.2014.

[3] Khronos Group. Webgl and opengl differences. Saatavissa `http://www.khronos.org/webgl/wiki_1_15/index.php/WebGL_and_OpenGL_Differences`. Viitattu 21.02.2014.

[4] Khronos Group. WebGL Specification, 2013. Saatavissa `http://www.khronos.org/registry/webgl/specs/latest/1.0/`. Viitattu 15.02.2014.

[5] R. C. Hoetzlein. Graphics performance in rich internet applications. *Computer Graphics and Applications, IEEE*, 32(5):98–104, 2012. Saatavissa IEEE Explore, DOI: 10.1109/MCG.2012.102. Viitattu 21.02.2014.

[6] Fredrik Smedberg. Performance analysis of javascript, 17.05.2010 2010. Saatavissa `http://www.diva-portal.org/smash/get/diva2:321661/FULLTEXT01.pdf`. Viitattu 21.02.2014.

[7] WHATWG W3C. HTML5 differences from HTML4: Introduction, 2011. Saatavissa `http://www.w3.org/TR/2011/WD-html5-diff-20110405/#introduction`. Viitattu 15.02.2014.

[8] WHATWG. HTML: 10 - Web workers, 21.02.2014 . Saatavissa `http://www.whatwg.org/specs/web-apps/current-work/multipage/workers.html`. Viitattu 21.02.2014.