|  |
| --- |
| Odisee |
| Project 2 |
| Android Applicatie |

|  |
| --- |
| Wouter Vande Velde  8-2-1995  Mentor: Mr. Demeester |

Contents

[Overzichtsrapport 2](#_Toc469915341)

[Gebruiksvriendelijkheid 2](#_Toc469915342)

[Ontwikkeling 2](#_Toc469915343)

[Programmeertaal 2](#_Toc469915344)

[Eindrapportering 4](#_Toc469915345)

[Reflectie 5](#_Toc469915346)

[Zelfevaluatie 6](#_Toc469915347)

[Bibliography 7](#_Toc469915348)

# Verslag

# LibGDX

LibGDX is een gratis en open-source framework om games te ontwikkelen, het werd gelanceerd begin 2014. Het sterkste punt van het framework is dat het toestaat om dezelfde code te gebruiken voor verschillende platforms zoals Linux, Windows, IOS, Android… Andere features van LibGDX zijn onder andere dat het gebruik maakt van OpenGL, 2D en 3D ondersteuning aanbiedt, … Alle Features zijn te lezen op de pagina ‘Goals and Features’ op de website van LibGDX [1] .

## Aanmaken project

Om het project aan te maken moet je een jar openen, deze jar opent een GUI waarop verschillende eigenschappen gekozen worden. Zo moet je de package naam geven aan het project en kiezen voor welke systemen het project ontwikkeld moet worden.

Eens het project aangemaakt is bevat het reeds een paar verschillende bestanden, de twee meest relevante bestanden zijn MyGdxGame en AndroidLauncher. MyGdxGame bevind zich in de core-module van het project, dit bestand zal de code bevatten voor het spel. De code in de core-module zijn uniform voor alle platformen en is de code die zal omgevormd worden om het spel te laten werken op alle platformen.

AndroidLauncher is de startklasse en zal de code in de core-module aanroepen.

**public class** AndroidLauncher **extends** AndroidApplication {  
 @Override  
 **protected void** onCreate (Bundle savedInstanceState) {  
 **super**.onCreate(savedInstanceState);  
 AndroidApplicationConfiguration config = **new** AndroidApplicationConfiguration();  
 initialize(**new** MyGdxGame(), config);  
 }  
}

## States

Het spel bestaat uit verschillende states, elke state is eigenlijk een ander scherm dat gebruikt wordt in het spel. De code van elke state is uitgerust met verschillende methodes, deze methodes zijn:

* Een constructor: de startmethode, variabelen initialiseren, …
* update: om berekeningen te doen en waardes te updaten.
* render: om het scherm te tekenen.
* dispose: om de textures te verwijderen om memory overflow te voorkomen.

De PlayState is de belangrijkste staat, hierin

# Google API

Een van de einddoelen van het project was om de gebruiker te laten inloggen met Google en een Google Leaderboard te tonen. Oorspronkelijk dacht ik dat dit 2 verschillende API’s zouden zijn, de Auth en de Games API. Bij het implementeren van deze 2 API’s zijn er veel obstakels in de weg gekomen maar deze zijn uiteindelijk allemaal overwonnen.

## Auth API

Deze API was oorspronkelijk geïmplementeerd in de applicatie, het gaf de gebruiker de mogelijkheid om in te loggen met zijn gewoon Google-account. Echter blijkt dat deze login niet gecombineerd kan worden met de Games API.

## Games API

# Overzichtsrapport

Het onderwerp van dit overzichtsrapport is applicaties op verschillende besturingssystemen. Er zal besproken worden wat de voordelen zijn van een bepaalde operating system en wat de nadelen zijn. Hoofdzakelijk zullen Android en IOS besproken worden in dit overzichtsrapport en hoe applicaties op deze besturingssystemen draaien.

## Algemeen

Een applicatie op IOS kan enkel verkregen worden via de App Store in ITunes terwijl een applicatie op Android verkregen kan worden via de Play Store, Amazon, … of je kan zelf een toevoegen. Dit heeft als voordeel dat er veel meer applicaties beschikbaar voor Android zijn terwijl de applicaties op IOS beter beheerd en gecontroleerd worden en dat deze dus zelden malware zullen bevatten.

In 2014 hadden 84.7% van de smartphones op de markt Android als besturingssysteem terwijl Apple maar 11.7% had. Dit betekent dat applicaties voor Android een veel groter doelpubliek zullen bereiken. Maar ongeacht dit grote verschil is het Apple die in 2014 nog de meeste inkomsten had via applicaties. [1]

**Vastellingen**

|  |  |  |
| --- | --- | --- |
|  | Android | IOS |
| Apps verkrijgen | Play store, Amazon,… | ITunes |
| Marktaandeel | 84.7% | 11.7% |
| Inkomsten | IOS-inkomsten 70% hoger dan Android | |

## Ontwikkeling

### Programmeertaal

IOS-applicaties worden geschreven in Swift, dit is een nieuwe taal geschreven door Apple speciaal voor IOS en OS X. Swift is nog een relatief nieuwe taal wat betekend dat het nog steeds volop in ontwikkeling is. Maar Swift is ook backwards-compatibel met zijn voorganger Objective-C wat betekend dat het toch veel libraries ter beschikking heeft. Een voordeel van deze taal is dat het een veilige taal zou zijn [2].

Android-applicaties worden geschreven in Java, het voordeel van deze taal is dat het op alle verschillende modellen van Android kan gedraaid worden. [3] Dit komt omdat Java gerund wordt in een virtual machine ongeacht de onderliggende architectuur. Een sterk punt van Java is dat het gebruik maakt van stack allocatie het heeft een automatische garbage collection Een groot nadeel van Java is dat het redelijk traag is, zowel in het compilen als het runnen.

**Vastellingen**

|  |  |  |
| --- | --- | --- |
|  | Android | IOS |
| programmeertaal | Java | Swift |
|  |  |  |

## Hardware

Bij sommige applicaties is het belangrijk dat de hardware die aanwezig is in het toestel de applicatie kan runnen. Het voordeel van IOS is dat het maar op 1 type device gerund wordt, de IPhone. Dit betekent dat de specificaties van het model enkel zullen verschillen onder de verschillende modellen. Dit is volledig tegengesteld bij Android, er zijn talloze verschillende toestellen die Android draaien met allemaal verschillende specificaties. De specificaties van een iPhone zijn goed waardoor je zeker kan zijn dat de applicatie gedraaid zal kunnen worden.

**Vastellingen**

|  |  |  |
| --- | --- | --- |
|  | Android | IOS |
| Modellen | Zeer veel verschillende | 1 type |
| Specificaties | Verschillen enorm | Optimaal |

# Eindrapportering

# Reflectie

# Zelfevaluatie

# Bibliography

|  |  |
| --- | --- |
| [1] | G. Sims, „Google Play Store vs the Apple App Store: by the numbers (2015),” AndroidAuthority, 20 april 2015. [Online]. Available: http://www.androidauthority.com/google-play-store-vs-the-apple-app-store-601836/. |
| [2] | P. Rubens, „10 Things You Should Know About Apple's Swift,” CIO, 21 juli 2014. [Online]. Available: http://www.cio.com/article/2456100/mobile-development/10-things-you-should-know-about-apples-swift.html. |
| [3] | M. Klaus, „Advantages and Disadvantages of Java Application Development,” Selfgrowth, [Online]. Available: http://www.selfgrowth.com/articles/advantages-and-disadvantages-of-java-application-development. |
| [4] | Google, „Start Integrating Google Sign-In into Your Android App,” Google, 26 oktober 2016. [Online]. Available: https://developers.google.com/identity/sign-in/android/start-integrating. |