Eindopdracht Android Game

# Opdrachtomschrijving

De eindopdracht is het ontwikkelen van een spel voor een Android apparaat, waarbij je gebruik maakt van de aangeleverde Android Phonegame Package. Deze opdracht maak je samen met een medestudent.

De opdracht is niet gespecificeerd. Met andere woorden, je bent vrij om je eigen spel te bedenken. Maak van het spel eerst een functioneel ontwerp en technisch ontwerp en bespreek dit met je docent voor je aan de uitwerking begint.

Bij deze eindopdracht moet je de volgende producten opleveren:

* Functioneel ontwerp (FO)
* Technisch ontwerp (TO)
* De broncode van de game met resources en api-documentatie

## Cijfer

Het functioneel en technisch ontwerp worden afgevinkt.

Voor het de game krijg je een cijfer mits FO en TO zijn afgevinkt.

## Functioneel Ontwerp

In het Functioneel Ontwerp staat in ieder geval het volgende:

* Naam van het spel.

Toad parcour (mario like game)

* Doel van het spel.

Zo veel mogelijk punten verzamelen in een gegeven tijdsperiode en je niet laten vermoorden door de monster/prinsessen.

* Korte omschrijving van de wereld waarin het spel speelt.

Het speelt zich af in een typische Mario wereld maar dan wel met windmolens, bananen en shrooms.

* Perspectief (van boven, of van opzij).

Opzij

* Beschrijving van acties die de (menselijke) speler kan uitvoeren.

Je kunt rondrennen, springen en Powers gebruiken. Ook kun je bepaalde objecten oppakken of een interactie mee uitvoeren.

* Beschrijving van objecten en obstakels die in het spel voorkomen, inclusief het gedrag van deze objecten.

Je kunt door monsters vermoord worden en de monsters vermoorden. Je kunt ook door gaten uit de wereld vallen, op dat moment ben je game over.

* Beschrijving van start en eind van het spel.

Je begint aan het begin de wereld. je kunt door de wereld heen lopen en punten verzamelen. Na een bepaalde tijd is het spel afgelopen en moet je in de de paddenstoel zitten, zo niet is het game over en worden je punten niet opgeslagen.

* Beschrijving van overige elementen (formulieren, dashboard, menu's) die in het spel een rol spelen.

Je hebt een dashboard waar je score op staat.

Formulier voor de naam van de speler

Dashboard voor de highscores

Hoofdmenu (alle opties van de game)

Moeilijkheidsgraad aanpassen.

* Beschrijving van het apparaat (specificatie van de telefoon, of tablet) waar de game in ieder geval op moet draaien.

Samsung Galaxy S4 + de emulator

* Schermschetsjes inclusief viewport (indien je er een gebruikt).

Zie old skool paper.

## Technisch Ontwerp

In het Technisch ontwerp (TO) is het volgende te vinden:

* UML klassendiagram met beschrijving van de klassen die in het spel voorkomen
* Een korte toelichting..

# Beoordelingsformulier

## FO

|  |  |
| --- | --- |
| Eis | Vink/Opmerkingen |
| Voldoende compleet. |  |
| Het document is voldoende gestructureerd en bevat nauwelijks taalfouten. |  |

## TO

|  |  |
| --- | --- |
| Eis | Vink/Opmerkingen |
| Voldoende compleet. |  |
| Het document is voldoende gestructureerd en bevat nauwelijks taalfouten. |  |

## Game

|  |  |
| --- | --- |
| Eis | Beoordeling/Opmerkingen |
| De code is voorzien van API documentatie voor alle eigen ontwikkelde publieke klassen en methoden. |  |
| Er zijn minimaal 8 eigen klassen gerealiseerd die voldoende verschillend zijn (klassen met vrijwel identiek gedrag met een verschillende klassenaam zijn niet voldoende). |  |
| Er wordt minimaal 1 van de bestaande interfaces uit de GamePackage geïmplementeerd en zinvol toegepast in de game. |  |
| Er wordt oververving toegepast door niet alleen van klassen uit de GamePackage over te erven, maar ook minimaal van een eigen klasse. Er wordt hierbij ook actief overriding toegepast (dus niet alleen voortborduren op het voorbeeldspel). |  |
| De studenten hebben een aantoonbare evenwichtige bijdrage (zo niet dan volgt een verschillend cijfer voor de teamleden), dit blijkt onder andere uit:   * De werkverdeling * De @author tag in de JavaDoc * Het beantwoorden van vragen tijdens de oplevering(en) * Mogelijk is een urenverantwoording gewenst |  |
| Klassen hebben een duidelijke verantwoordelijkheid   * Naam van de klasse komt overeen met de taak * Namen van de attributen en methoden dekken de lading * Er wordt zo min mogelijk dubbele code gebruikt. * Geen static variabelen, tenzij je daar een goede reden voor hebt |  |
| Datastructuren uit de course worden toegepast   * ArrayList zodra dit zinvol is |  |
| Studenten tonen mondeling aan dat ze hun eigen code begrijpen. |  |
| (Optioneel) Extra’s |  |