

# Inhoudsopgave

Inhoud

[Inhoudsopgave 2](#_Toc354874548)

[Samenwerkingscontract 4](#_Toc354874549)

[Project beschrijving 5](#_Toc354874555)

[Gameplay eisen 6](#_Toc354874557)

[Technical design 9](#_Toc354874569)

[Graphics Design 10](#_Toc354874575)

[Verantwoordingsdocument 12](#_Toc354874584)

[Agenda 13](#_Toc354874585)

[Urenverantwoording 14](#_Toc354874586)

[Hand-out presentatie 18](#_Toc354874587)

# Samenwerkingscontract

## Aanwezigheid

* Zorg dat je op tijd bent. (zie Sancties 1)
* Als iemand zonder reden afwezig is volgen er sancties. (zie Sancties 2)
* Bij regelmatige overtredingen wordt er actie ondernomen. (zie Sancties 3)
* Als je van tevoren weet dat je niet aanwezig bent, meldt dit. (zie Sancties 4)

## Inzet

* Iedereen moet alles op tijd inleveren. (zie Sancties 2)
* Wij verwachten ook een hoge kwaliteit bij wat je inlevert. (zie Sancties 5)

## Sancties

1. Bij twee keer te laat komen na de afgesproken vijftig minuten volgt er een gele kaart.
2. Bij het te laat inleveren van het afgesproken werk, of de afspraken niet nakomen, volgt er een gele kaart.
3. Na twee gele kaarten stapt de groep naar de projectbegeleider, met kans op verwijdering uit de groep.
4. Meld je 6 uur van te voren als je niet kan komen.
5. Als de groep niet met het afgeleverde werk tevreden is, dient diegene zijn werk te verbeteren.

## Werkwijze

* In groepsverband op school tijdens de beschikbare tussenuren.
* Thuis werken en communiceren via Skype, e-mail en de Google Documents map die gedeeld is voor ieder project lid.

## Contact

|  |  |
| --- | --- |
| Johnny van den Berg  0816348  E-mail: [johnnyvdberg@gmail.com](mailto:johnnyvdberg@gmail.com)  Skype: johnnyvdberg  06-15580881 | Davey van Nes  0823826  E-mail: [dave.pirate@gmail.com](mailto:Dave.pirate@gmail.com)  Skype: kapytanhook  06-22038499 |
| Martijn Heurter  0816127  E-mail: [martijn.heurter@gmail.com](mailto:martijn.heurter@gmail.com)  Skype: stardarknl  06-31932460 | Hrvoje Miljak  0819516  E-mail: [miljak.hrvoje@gmail.com](mailto:miljak.hrvoje@gmail.com)  Skype: miljaker  06-53481301 |

Handtekening Johnny van den Berg Handtekening Davey van Nes

Handtekening Martijn Heurter Handtekening Hrvoje Miljak

# Project beschrijving

Asshole Asteroids speelt zich af in de ruimte, waar de speler een asteroïde bestuurt die door hetuniversum beweegt. Zijn taak is om andere objecten in zijn weg te vernietigen. De asteroïde kan een ander object vernietigen aan de hand van verschillende factoren, deze zijn snelheid, hoek van impact en de grootte van het object.

Een object zou een andere asteroïde kunnen zijn, maar ook planeten, planetoïden, sterren en ruimteschepen horen hierbij. Het doel is om zoveel mogelijk punten te verzamelen die de speler krijgt voor het vernietigen van objecten. De speler krijgt een audiovisuele beloning als hij een effectieve ‘asshole’ is.

# Gameplay eisen

## Speler

* De speler stuurt een asteroïde aan.
* Het punt van kijken is vanuit de derde persoon.
* De asteroïde kan zich beperkt bewegen in het speelveld.
* De tijd verloopt een stuk langzamer als de speler een planeet kan raken.

## Andere objecten

De speler zal meerdere objecten in het speelveld tegenkomen.

De interactie tussen deze objecten en de speler verschilt per type object.

### Planeet

* Willekeurig gegenereerd
  + Willekeurig type
  + Willekeurige grootte
  + Willekeurige populatie
  + Willekeurige naam
* Kan vernietigd worden door de speler
  + Planeet moet in stukken breken op basis van de impact van de asteroïde op de planeet
  + De impact kan de asteroïde vertragen op basis van grootte van de planeet
* Drie soorten planeten
  + Vuurplaneet
    - Gevoelig voor ijs powerup
  + Dode planeet
    - Kans op kleine onderzoekingspopulatie op planeet
  + Zwavel planeet
  + Leven ondersteunende planeet
    - Grootte bepaalt bevolking
  + IJs planeet
    - Gevoelig voor vuur powerup
  + Gas planeet
    - Niet vernietigbaar door speler

### Sterren

* Per speelveld één ster
* De zwaartekracht van de ster versnelt de asteroïde
* Onverwoestbaar
  + Als de speler in aanraking komt met de ster zal het spel over zijn

### Asteroïde

* Vast model
  + Willekeurige grootte
  + Willekeurige texture
* Vernietigbaar
* Kleiner dan de asteroïde van de speler
* Vertragend indien aangeraakt wordt door speler

## Game modes

Er zijn verschillende game modes die de speler in het menu kan kiezen.

### Arcade

* Speelveld
  + Willekeurig gegenereerd met objecten die in het vorige hoofdstuk beschreven staan
  + Heeft een vaste grootte
* Power-ups
  + + Snelheid: Actief (Speler moet zelf activeren)
  + + Grootte: Passief (Power-up wordt meteen toegepast)
* Score-systeem
  + Objecten vernietigen
  + Natuur van het object bepaald de score
  + De vernietiging zelf bepaald ook de score
  + Snelheid van de asteroïde is vermenigvuldiger
  + Willekeurige planeten hebben bevolking, die bonus punten geven
* End-game
  + Na een tijd wordt de moeilijkheidsfactor te groot (planeet groottes) waardoor het bijna onmogelijk word om te overleven.

### Time-attack

* Zelfde regels als Arcade
* Moeilijkheidsfactor wordt hoger over tijd
* Tijdlimiet (3 minuten)
* Power ups
  + +Tijd (mogelijk)

### Slingshot

* Gescripte map
* De speler maakt gebruik van sterren om snelheid te krijgen
* De speler moet planeten vernietigen om het level te halen

## Besturing

De speler zal de indruk krijgen dat hij vooruitgaat, terwijl het eigenlijk de wereld om hem heen is dat naar hem toekomt. De besturing zal instelbaar zijn.

Beweging: Pijltjestoetsen (standaard)

Actie toets: Spatie (standaard)

Escape toets: Naar menu (niet instelbaar)

## Interface

De speler kijkt altijd voor zich uit. De speler kan informatie over tegemoet komende objecten zien als de speler zich binnen een bepaalde afstand van het andere object bevind. Dit is een relatief kleine afstand. De informatie van andere objecten wordt in een ‘Terminator-style’ getoond. Dit wil zeggen dat er een pijltje naar het object wijst met daaraan een informatieveld waarin de benodigde informatie getoond word. De informatie die getoond zal worden verschilt per object. Er kan bijvoorbeeld in staan (als het toepasbaar is) hoe groot het object is, hoeveel inwoners het heeft, wat de score vermenigvuldiger zal zijn, etc. De informatie zal kort beschikbaar zijn, en alleen als je, als speler zijnde, naar het ander object toe beweegt. Er is een interface, die verder beschreven wordt in het hoofdstuk Graphics Design.

# Technical design UDK

UDK word in het eindproduct niet meer gebruikt, echter is het wel deel geweest van het project en veel van onze beslissingen in UDK konden ook overgezet worden naar HTML5. Aangezien het doel van ICT-Lab meer om het experimenteren dan het eindresultaat gaat, blijft deze documentatie in het eindverslag.

## Engine

Er word gebruik gemaakt van Unreal Engine 3, die bij de Unreal Development Kit zit. In de UDK kunnen functies gescript worden met UnrealScript. Deze engine is gratis voor non-commerciële doeleinden op Windows.

Er kan uitgerold worden naar meerdere platforms, op het moment concentreren we alleen op Windows 7 en 8.

## Map beperkingen

Om de illusie van een groot heelal te creëren bewegen de objecten naar de speler toe, in plaats van de speler naar het object. De mappen worden anders veels te groot en te statisch. Door de positie en hoeken van de skybox en de modellen dynamisch aan te passen, lijkt het alsof de mappen ongelimiteerd groot zijn. Doordat de speler zo snel lijkt te gaan en planeten vergeleken met de ster als lichtbron zo klein, is het niet onrealistisch als de eerdere planeten of objecten niet meer te zien zijn na een korte tijd periode.

## Destruction physics

Er kan gebruik gemaakt worden van de volgende manieren om destruction physics toe te passen.

### Havok

Dit is een erg realistische maar zware manier van destructie. We hebben het uitgetest maar het lijkt meer werk om dit in een game te implementeren dan PhysX.

### PhysX

Dit is een Nvidia manier om met destruction om te gaan. Het voordeel hieraan is dat Nvidia hardware het versneld. Echter kan het met andere grafische kaarten ook gedraaid worden, alleen dan via de CPU. Ook zit het in veel engines al ingebouwd. Net als bij Unreal.

Om dit te gebruiken kan je modellen opsplitsen in kleinere delen en aangeven wat de massa en draagkracht is. Dit gaat door middel van tools uit de APEX destruction toolchain.

Al bij al levert dit een zeer realistisch resultaat op.

# Technical design HTML5

Na veel overleg werd het duidelijk dat UDK niet haalbaar was voor wat het team wilde neerzetten. Daar UDK vooral gefocust was op First Person Shooters en ons product technisch teveel daarvan afwijkt.

Na het besluit was genomen om UDK niet als engine te gebruiken was het de vraag wat de taken van UDK zou moeten vervangen. Nadat bleek dat er te weinig kennis en vooral weinig tijd overbleef om een andere 3d engine te implementeren, is er gekozen voor HTML5. Voordelen zijn de multi-platform compatibility en de kennis die in het team beschikbaar is. Bovendien is in de afgelopen tijd significante vooruitgang geboekt in de snelheid en toepasbaarheid van HTML5 Canvas in combinatie met Javascript. Genoeg mogelijkheden om te experimenteren dus. Nadelen zijn echter nog steeds de snelheid ten opzichte van conventionele engines gemaakt met C++ of andere talen, ondanks alle vooruitgang, waardoor het spel niet heel zwaar kan worden.

## Engine

Aangezien het team veel ervaring heeft met HTML en Javascript of soortgelijke talen, heeft het team besloten voor een eigen engine van de grond op. De redenen hiervoor waren simpel: Minder afhankelijkheid van derde partijen en meer controle over elk aspect van het spel.

## Beperkingen en oplossingen

Doordat er veel tijd gaat zitten in aspecten nabouwen die in UDK juist erg simpel waren, moest er nagedacht worden over de meer complexe onderdelen en of deze echt cruciaal voor de game waren.

### Map simulatie

In UDK hadden het team het idee om objecten naar de asteroïde toe te laten komen in plaats van andersom. Echter kan dit in HTML5 veel beter toegepast worden met 2D technieken. Aangezien de speler toch de planeten van één kant ziet kan statische belichting in de planeten worden toegepast. Naarmate de asteroïde dichterbij komt, kan de illusie door een plaatje te vergroten gedupliceerd worden.

Maar alleen het dichterbij laten komen van plaatjes geeft niet het gewenste effect. Om het meer te laten lijken alsof de speler een asteroïde is die door het zonnestelsel zweeft, zijn er ‘orbit’ effecten toegevoegd. Het idee is dat de asteroïde om de zon draait, maar nooit de zon zal zien omdat deze nooit recht voor de speler zit. Het draaien rondom de zon wordt op twee punten gerealiseerd. Ten eerste draait de sterrenachtergrond mee op basis van snelheid, zodat het echt lijkt alsof de speler zich in een baan bevind. Ten tweede wordt er aan de linkerkant van het scherm een gloeiend plaatje geplaatst. Als de speler ‘dichter’ bij de zon komt, dan zal dit effect toenemen. Als de speler zich er erg ver vanaf bevind, dan zal het effect compleet verdwijnen. Het resultaat is dat het lijkt alsof er zich een zon in het midden van de map bevind, waar je langs zweeft.

Om het gevoel van snelheid te versterken, worden naast de background ook kleine particles over de asteroïde heen getekend. Die laten een spoor achter die sneller of langzamer gaat naarmate de speler snelheid opbouwt.

### Planeet generatie

Het team wilde in het begin compleet willekeurige planeten, met andere textures en eigenschappen. Echter zou dit veel rekenkracht kosten voor de gebruiker. Het alternatief is meerdere planeten toevoegen. Niet willekeurig qua uiterlijk, maar wel qua eigenschappen.

Er zijn verschillende typen planeten, elk met een paar textures die willekeurig toegewezen worden. Deze planeet typen hebben een paar eigenschappen: Kans op bevolking, grootte en score vermenigvuldiger. Als er een planeet gegenereerd wordt, dan worden deze eigenschappen gebruikt om een unieke planeet te maken. De bevolking, grootte en score zal willekeurig variëren per planeet. Zo zal de ene planeet toch andere eigenschappen hebben in vergelijking met de andere.

De locatie van de planeet word opgeslagen als een tweedimensionaal gegeven, afstand van de zon en de hoek ten opzichte van de bovenkant. Met wat goniometrie kan de daadwerkelijke x en y coördinaat worden bepaald. Op basis van het planeet type word ook de positie gebaseerd. Zo zal een reusachtige gasplaneet nooit dicht bij de zon komen te staan, en een bevolkte planeet dichter bij de zon dan een vuurplaneet. De hoek word echter wel compleet willekeurig gehouden, mits deze niet botst met een andere planeet.

### Planeet vernietiging

In tegenstelling tot PhysX die in UDK beschikbaar was, heeft HTML5 natuurlijk qua realisme niet veel te bieden. Echter was het team besloten om alsnog een effect toe te voegen die het vernietigingsgedeelte van het spel alsnog grafisch aangenaam te maken. Hierbij kwam het team uit op een combinatie van particles en een kleine animatie.

Als de speler de planeet raakt, dan worden er een aantal (op grafische instelling ‘high’ 1000) objecten aangemaakt met eigen coördinaten, snelheid en leeftijd. De snelheid word aangepast op basis van het raakpunt met de planeet, dat wil zeggen: Als de planeet van links geraakt word, dan spuiten de particles naar rechts en andersom. Deze snelheid varieert per particle om het wat meer realisme te geven. De leeftijd word ook willekeurig bepaald, zodat alle particles niet op het zelfde moment verdwijnen. Tenzij de particle buiten het scherm valt, omdat de speler het dan toch niet meer kan zien.

Naast de particles is er ook nog een statische animatie die een ring laat zien, om het iets complexer en realistischer eruit te laten zien.

# Graphics Design

## Menu



### Hoofd menu

Een achtergrond die het ruimte-thema van de game aangeeft. Met de game modes en opties als transparante knoppen. De achtergrond wordtgeanimeerd.

### Geluid

Er wordt in de achtergrond rustige muziek gespeeld om de sfeer van de ruimte te benadrukken. Daarnaast is er hoorbare feedback als er over een knop gezweefd of geklikt word.

### Opties

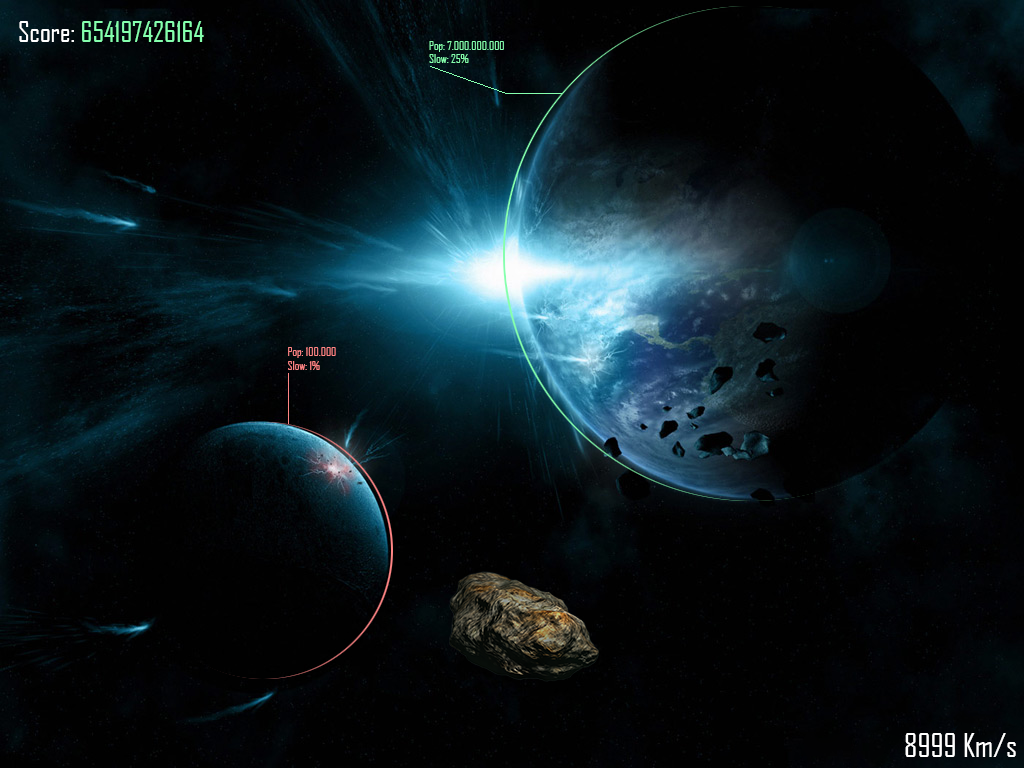
In dezelfde stijl als het menu worden er nog te definiëren opties beschikbaar gesteld. De opties zullen in een tabelvormig menu weergegeven worden, waarbij elk niveau verder in het menu een kolom toevoegt.

### Quit

Toont een bevestigings-popup met de vraag of je daadwerkelijk het spel wilt afsluiten en terugkeren wilt keren naar je desktop.

## In game

Mock up van het speelveld waar naar gestreefd word.



### Speler

De asteroïde die de speler bestuurd wordt onderaan, in beeld gebracht en zal daar blijven gedurende het spel. Doormiddel van de pijltjestoetsen kan de asteroïde tijdelijk naar links of naar rechts, maar zal snel weer teruggaan naar het midden.

### Andere objecten

Andere objecten zoals planeten zijn ook in beeld gebracht. Door middel van de ‘Terminator’-view (eerder beschreven in het hoofdstuk ‘Gameplay design’) kan de benodigde snelheid en verwachte punten voor de vernietiging van een object weergegeven worden.

### Overige informatie

Als extra visuele indicatie voor het succes van het rammen van de planeet, wordt de informatie in groene of rode tekst weergegeven.

### Muziek en geluid

Als het spel begint hoort de speler direct energieke muziek. Er zijn enkele nummers zodat de speler niet telkens hetzelfde nummer hoort. Naast de muziek komt er een alarm geluid als een asteroïde de speler bijna raakt en een ‘duivelse’ stem als de speler iets goed of fout doet.

### HUD

Links onderin wordt de score van de speler bijgehouden die hij heeft behaald in de huidige poging. Onder de score worden enkele statistieken bij gehouden, waarbij ‘Time Left’ alleen relevant is in de ‘Time-Attack’ spelmodus.

Opvallend is de mini map, die cruciaal voor het spelen is. De ronde witte lijn stelt de baan rond de zon voor, de groene stipjes zijn de planeten die de speler kan raken, en de asteroïde is de speler zelf. Als de witte baan rood wordt, dan weet de speler dat hij de kans heeft om de planeet te raken. Zonder de mini map is het lastig om de planeet te vinden, omdat het zonnestelsel zo groot is.

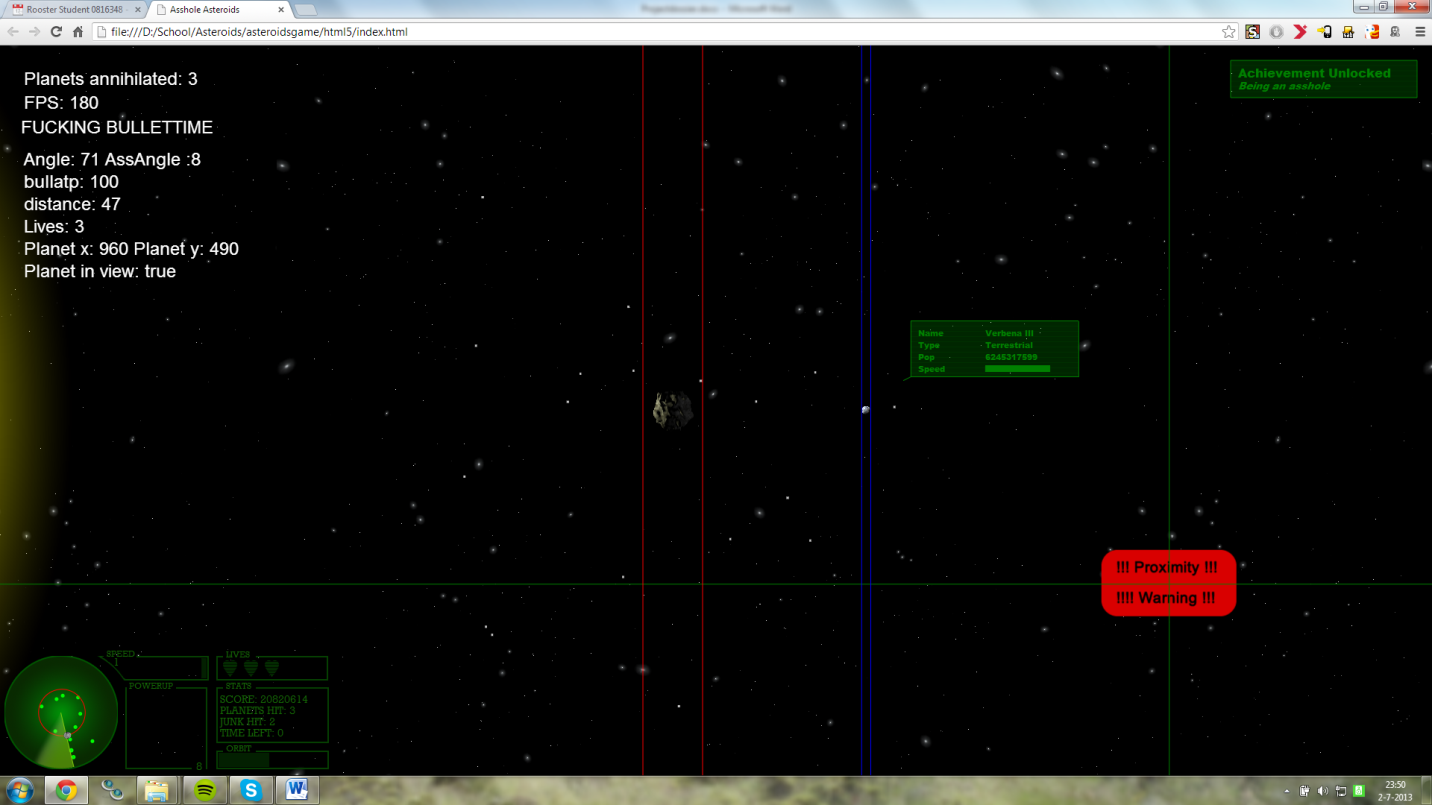
Naast de mini map word de snelheid van de speler bijgehouden, waarmee de speler kan bepalen of de planeet geraakt kan worden zonder zelf dood te gaan. Als de speler dood gaat dan gaat er een hartje af van de Lives kolom. De ‘orbit’ balk geeft aan hoe ver de speler van de zon af is, te dichtbij en de balk wordt rood, te veraf en de balk wordt ook rood.

De powerup kolom geeft aan welke bonus de speler tot zijn beschikking heeft met een timer hoelang deze nog beschikbaar is.



### Planeet informatie

Wanneer de speler een planeet zou kunnen zien, maar de planeet nog te ver af is, krijgt de planeet een indicator. Deze indicator geeft snel de belangrijke informatie over de planeet. Heeft de speler genoeg snelheid om de planeet te raken, dan zal de speed balk vol zijn. Als de planeet bevolkt is, dan krijgt de speler een idee hoeveel extra punten hij krijgt. Het planeettype maakt het in het begin van het level makkelijker om te bepalen hoeveel snelheid er ongeveer nodig zal zijn, en of er een powerup voor beschikbaar is.



# Sound Design

# Gebruikte tools

## Bewerken van code

Adobe Dreamweaver CS6

Notepad++

Microsoft Visual Studio 2010

## Grafische tools

Maya

GIMP

Adobe Photoshop CS6

Spacescape  
Adobe Fireworks CS6

## Browsers

Chrome 27

Internet Explorer 10

## Geluidsbewerking en opname

Audacity

Martijn?

# Verantwoordingsdocument

Essentiële Globale Taken

|  |  |  |
| --- | --- | --- |
| **Taak** | **Eind verantwoordelijke(n)** | **Voortgang** |
| Gameplay uitwerken | Iedereen | 80% |
| Modeleren | Hrvoje | 20% |
| Design | Hrvoje | 5% |
| Graphics | Hrvoje | 5% |
| Scripten UDK | Johnny / Martijn | 0% |
| Globale Physics | Johnny | 5% |
| Maps Maken UDK | Martijn | 0% |
| Apex Destructables | Davey | 10% |
| Maps cooken UDK | Davey | 5% |
| Documentatie ICTLAB01 | Iedereen | 90% |
| Documentatie ICTLAB02 | Iedereen | xx% |

# Agenda

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  | **ICTLAB01** | |  |  |  |
| Week 6 |  | Week 7 |  | Week 8 |  | Week 9 |  | Week 10 |
|  |  |  |  |  | Presentatie | | Documentatie | |
|  |  |  |  |  |  |  | opleveren | |
|  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |  |  | **ICTLAB02** |
|  | Week 11 |  | Week 12 |  | Week 13 |  | Week 14 |  | Week 15 |
|  |  | Prototype | |  |  | Prototype | |  |  |
|  |  | opleveren | |  |  | opleveren | |  |  |
|  |  |  |  |  |  |  |  |  |  |
| Sprint 1 | |  |  |  | Sprint 2 | |  |  | Sprint 3 |

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  |  |  |  |  |  |  |  |
|  | Week 16 |  | Week 17 |  | Week 18 |  | Week 19 |  | Week 20 |
| Prototype (alpha) | |  |  | Prototype (beta) | | Release candidate | | Eindproduct | |
| opleveren | |  |  | opleveren | | opleveren | | opleveren | |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  | Sprint 4 | |  |  |  |  |  |

|  |  |  |  |
| --- | --- | --- | --- |
| **Legenda** |  |  |  |
| Onderzoeksfase | | |  |
| Experimentelefase | | |  |
| Realisatiefase | | |  |
| Eindfase | |  |  |

# Urenverantwoording

Davey van Nes

|  |  |  |
| --- | --- | --- |
| **Week** | **Omschrijving** | **Uren** |
| 2 | Start ICTLAB, Brainstormen | 10 |
| 3 | Veel discussiëren, veel brainstormen. | 8 |
| 4 | Concept vastgesteld, begonnen aan documentatie | 10 |
| 5 | Concept art | 3 |
| 6 | Proberen van Havok Destruction engine + Unity 3D + Torque game engine | 10 |
| 7 | Test van Apex destructables + Unreal Development kit | 5 |
| 8 | Importeren modellen in Apex | 2 |
| 9 | Maken presentatie en verder uitwerken concept | 8 |
| 10 | Documentatie KW3 | 7 | 57 |
| 11 | Unreal Kismet, Asteroid model instellen als voertuig, camera vastknopen in 3e persoon | 10 |
| 12 | Laatste pogingen unreal, zwaartekracht en character editen en begin html check | 10 |
| 13 | Overstap op html, Canvas onderzoek, webgl onderzoek, basis spacegame opzetten, loader testen, redesign van game concept | 15 |
| 14 | Html 5 Canvas Draw Scaled , draw rotated, speedtesting | 12 |
| 15 | Menu, animaties, muis hover effect, hele game engine omgooien | 14 |
| 16 | Planeten spawnen, collisie, explosie op basis van positie, particle effects | 12 |
| 17 | Betere achtergrond, Snelheden, bullettime, realistische orbit | 12 |
| 18 | Animaties astroid, in autodesk, explosies als scalende sprites | 7 |
| 19 | Tweaks en bugfixes, gameplay elementen toevoegen. | 6 |
| 20 | Documentatie, 3000 bugfixes, paar sounds erin, score dingen | 15 |
| Totaal |  | 113 | 170 |

Johnny van den Berg

|  |  |  |  |
| --- | --- | --- | --- |
| **Week** | **Omschrijving** | **Uren** | |
| 2 | Start ICTLAB, Brainstormen | 10 | |
| 3 | Veel discussiëren, veel brainstormen. | 7 | |
| 4 | Concept vastgesteld, begonnen aan documentatie | 8 | |
| 5 | Mogelijkheden onderzocht van game engines en physics, testtaken verdeeld | 9 | |
| 6 | Proberen van PhysX engine en UDK | 8 | |
| 7 | Unreal Development kit - experimenteren met physics (0g, simpele damage models en collision) | 7 | |
| 8 | Unreal Development kit - verder experimenteren met physics voor 0g | 8 | |
| 9 | Maken presentatie en verder uitwerken concept | 6 | |
| 10 | Documentatie KW3 | 8 | 71 |
| 11 | Skybox graphics | 5 | |
| 12 | Proberen objecten statisch in de wereld te zetten | 6 | |
| 13 | Overstap naar HTML5, mogelijkheden onderzoeken en experimenteren | 8 | |
| 14 | Aan particle systeem systeem gewerkt, plaatjes gemaakt voor menu | 11 | |
| 15 | Particle systeem afgemaakt en speed warp effect grotendeels werkend | 10 | |
| 16 | Speed warp toegevoegd in game en begonnen met minimap | 9 | |
| 17 | Verder met minimap, heeft nu orbit, collision en ‘radar’ sweep | 10 | |
| 18 | Planeet indicator en star glow gemaakt. Nieuwe planeet types toegevoegd: bevat eigenschappen zoals naam, populatiekans en planeettype | 11 | |
| 19 | Compatibility fix voor IE10, nieuwe gameplay sounds, score systeem toegevoegd. HUD updated voor alle relevante info | 12 | |
| 20 | Game polijsten en documentatie | 9 | 91 |
| Totaal |  | 162 | |

Martijn Heurter

|  |  |  |
| --- | --- | --- |
| **Week** | **Omschrijving** | **Uren** |
| 2 | Start ICTLAB en bedenken ideeën | 10 |
| 3 | Veel ideeën bedenken | 7 |
| 4 | Concept tot stand gebracht en besproken | 8 |
| 5 | Onderzoek naar eventueel te gebruiken engines | 9 |
| 6 | Proberen van PhysX engine en UDK. Gameplay beschreven. | 8 |
| 7 | Unreal Development kit - experimenteren met physics | 7 |
| 8 | Unreal Development kit - verder experimenteren | 5 |
| 9 | Maken presentatie, presenteren en verder uitwerken concept | 6 |
| 10 | Documentatie ICTLAB01 | 12 |
| Totaal |  | 70 |

Hrvoje Miljak

|  |  |  |
| --- | --- | --- |
| **Week** | **Omschrijving** | **Uren** |
| 2 | Start ICTLAB en bedenken ideeën | 10 |
| 3 | Veel ideeën bedenken en brainstormen | 8 |
| 4 | Andere ideen bedenken en asshole asteroids uitwerken | 12 |
| 5 | Mogelijkheden onderzocht van game engines en physics | 10 |
| 6 | Bekend raken met UDK en tutorials kijken. | 7 |
| 7 | Bekend raken met Maya en 3ds max, leren hoe modelen te maken. | 12 |
| 8 | Test modelen maken en exporteren om destructie te laten werken. | 6 |
| 9 | Maken presentatie, presenteren en verder uitwerken concept | 6 |
| 10 | Documentatie KW3 | 6 |
| Totaal |  | 77 |

# Hand-out presentatie