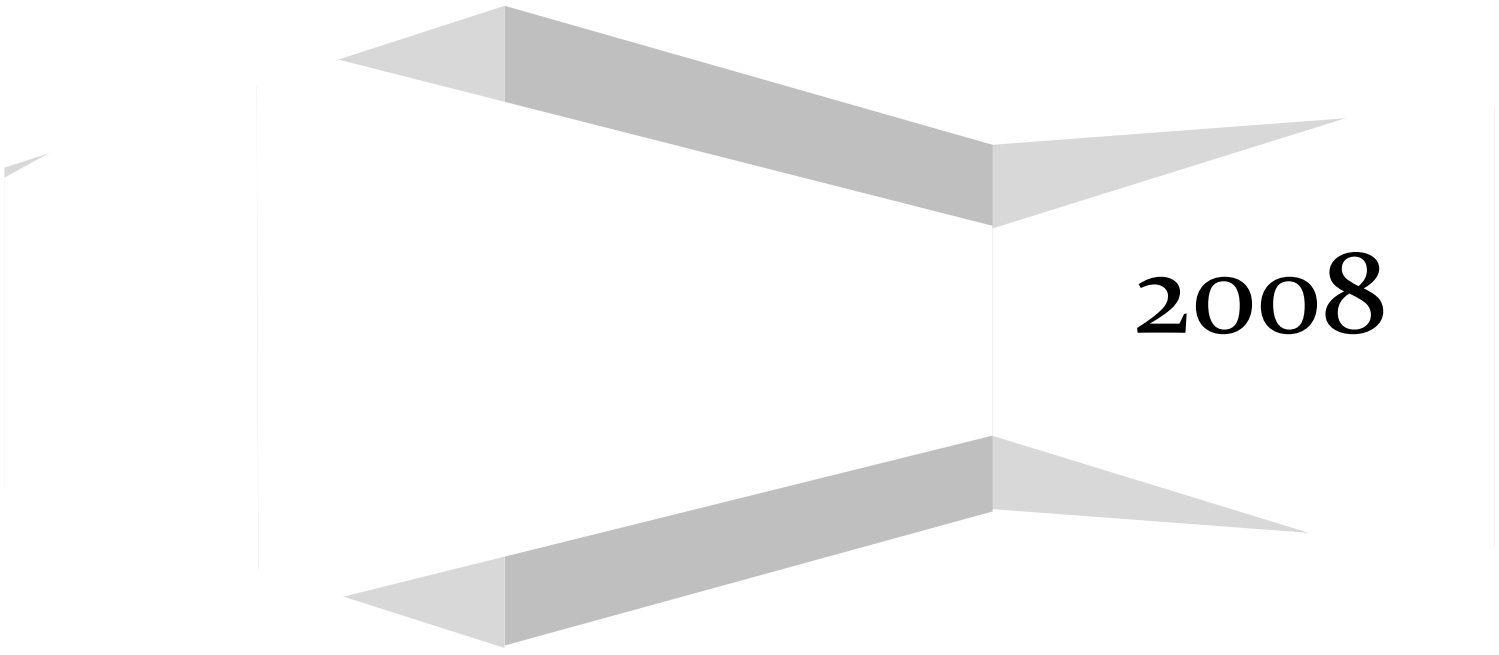
HJMenu Project

Scriptie

Paul Nijssen

Nr. 1502590

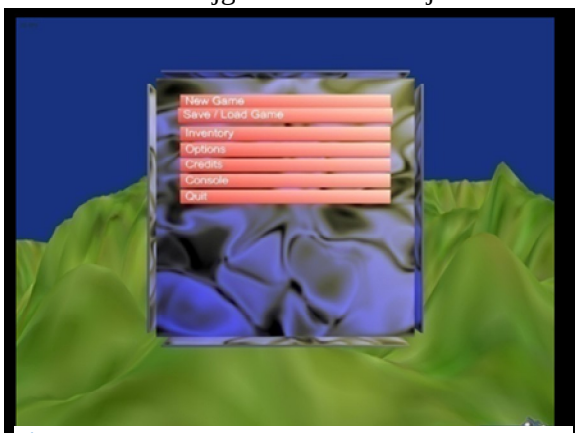


2008

1 Inhoud

2 Voorwoord

Dit project zal onderdeel worden van een groter project. Een volledig 3D computerspel. Plannen om een volledig 3D spel te creëren zijn ontstaan tijdens een eerder project tijdens mijn studie Technische Informatica aan de Hogeschool Utrecht. Omdat een project van deze grootte enkele jaren in beslag gaan nemen, is gekozen een afzonderlijk deel te realiseren dat als vooronderzoek kan dienen. Met dit project, het maken van het menu, is geprobeerd een beter inzicht te krijgen in de werkwijze en structuur van een 3D omgeving.



Figuur 1

Het menu waar dit project zich op zal richten zal in een 3D omgeving worden gerealiseerd. Dit betekend dat niet alleen lengte en breedte, maar ook diepte van belang is voor de presentatie van de onderdelen.

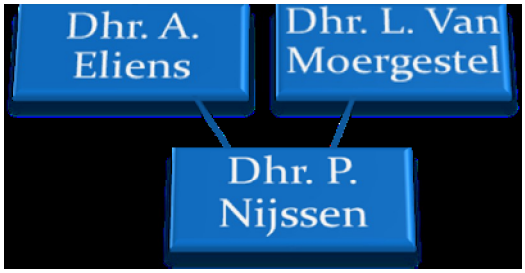
Dit document zal zich verder richten op het verloop en het resultaat van het project. Het eerstvolgende hoofdstuk zal de managementsamenvatting zijn, dit geeft een kort en bondig overzicht van de gehele scriptie. Het heeft als doel om in een korte tijd het verloop te bespreken zodat een goed beeld kan worden gevormd van het project zonder de volledige scriptie te lezen. Voor gedetailleerde informatie is dit hoofdstuk echter niet, hiervoor dient het de volledige scriptie geraadpleegd te worden.
Woorden die zijn aangegeven met een '*' worden verduidelijkt in de woordenlijst achterin dit document.

<<< korte uitleg alle hoofdstukken >>>

3 Managementsamenvatting

4 Organisatie

In dit hoofdstuk zal de projectorganisatie worden besproken. Een beschrijving van de betrokken personen en de wijze van communicatie.



Figuur 2

Het project zal worden gerealiseerd binnen de Hogeschool Utrecht, onder begeleiding van Dhr. L. Van Moergestel en in opdracht van Dhr. A. Eliens, werkzaam aan de Vrije Universiteit van Amsterdam.

Dhr. A Eliens fungeert als opdrachtgever en zal op- en aanmerkingen leveren over de tussentijdse voortgang en het uiteindelijke resultaat.

Dhr. L. Van Moergestel fungeert als begeleider bij het projectproces. De nadruk ligt hierbij op het controleren van de voortgang en beoordeling van de documentatie.
De uitvoerende, P. Nijssen, is verantwoordelijk voor het uiteindelijke resultaat. De rol van projectleider en projectlid zijn beide van toepassing. Er zijn geen andere projectleden in de organisatie.

De communicatie en informatievoorziening heeft plaatsvinden via e-mail. Bij grotere voortgang en belangrijke tussenproducten is een korte demonstratie gegeven.

5 Opdrachtomschrijving

Dit hoofdstuk richt zich op de opdracht, het maken van een 3D menu. Om duidelijkheid in de opdracht te krijgen is het verstandig de functie van een menu te verduidelijken. Een menu dat gebruikt word binnen een computerspel zal de functie hebben van het weergeven van opties, zoals de mogelijkheid een nieuw spel te starten of de inventaris van de speler te bekijken. Ook geeft een menu de mogelijkheid instellingen van het spel te wijzigen.

5.1 Doelstelling

Het maken van het 3D menu heeft als doel bekend te raken met de Ontwikkelomgeving*, het geven van meer flair aan een anders zeer statisch onderdeel van een computerspel en het bepalen van een overzichtelijke structuur voor het vervolgproject.
Het project functioneert ook als try-out voor het vervolgproject, wanneer dit project met succes afgesloten kan worden geeft dit aan dat de omgeving en de kennis aanwezig is om een stap verder te gaan en verdere verdieping in de ontwikkeling van 3D software te bewerkstelligen.

5.2 Probleemstelling

Bij het begin van dit project bezit de uitvoerende weinig kennis van een 3D omgeving. Het is bekend dat een engine* methoden bezit voor het tonen van objecten. De aanroep en aansturing van deze methoden is onbekend.
Enkele vragen die bij aanvang moeten worden gesteld zijn;

- Wat is de structuur van de engine?
- Wat zijn de mogelijkheden van een spelmenu?
- Welke mogelijkheden moeten worden gerealiseerd in het menu?

5.3 Opdracht

Ontwerp een 3D menu voor een computerspel met minimaal de volgende functionaliteit;
het moet mogelijk zijn een nieuw spel te starten (de methode moet aanwezig zijn), het moet mogelijk een spel te laden en op te slaan en er moet een inventaris weer te geven zijn.
Het ontwerp moet ook een aantal animaties bevatten. Denk hierbij aan een opstart- en afsluitanimatie, alsmede tussentijdse korte animaties tussen de verschillende menuonderdelen.

5.4 Eindproduct

Het eindproduct is het bovengenoemde menu, compleet met animaties en functionaliteit, onderdelen van het menu zullen in het hoofdstuk producten nader worden besproken. Verder dient er documentatie te worden geschreven, uitleg van de broncode en gemaakte keuzen, een scriptie (dit document) en een eindpresentatie. De eindproducten zullen later in dit document uitgebreid worden behandeld.

6 Uitgangssituatie

Dit hoofdstuk zal de volgende zaken behandelen;

- De beginsituatie, welke onderdelen zijn voorhandig bij het starten van het project.
- De verwachtingen, welke verwachtingen heeft de projectgroep bij aanvang van het project.

6.1 Beginsituatie

Bij het begin van het project zijn de volgende onderdelen beschikbaar.

Ontwikkelomgeving:

Visual Studio
3D engine

Met behulp van het ontwikkelprogramma Visual Studio kan op een overzichtelijke wijze het programma worden gemaakt en getest. De 3D engine levert een functionaliteit waarmee objecten (meshes*) kunnen worden weergegeven en bewerkt.

Verder gebruikte software

Microsoft Office
3D Studio Max
Adobe Photoshop

Office is een programma voor het schrijven van documentatie en maken van schema's.
Het programma 3D Studio Max is bedoeld voor het maken van modellen. Een model kan een mesh (statisch object) of een actor* (bewegend) zijn. Een gemaakt model kan worden opgeslagen als mesh of actor en door de engine worden geladen.
Photoshop is een programma voor het maken en bewerken van grafische bestanden. Bestanden die hiervoor in aanmerking komen zijn texturen en eventueel een heightmap*.

6.2 Verwachtingen

Verwachtingen voor dit project bij aanvang zijn positief. Dat wil zeggen dat de projectgroep voldoende mogelijkheden ziet om het doel te realiseren met de aanwezige materialen. Hoewel dit een nieuwe omgeving is, is de verwachting dat de noodzakelijke onderdelen binnen de tijd te realiseren zijn. De noodzakelijke en eventuele producten zullen in het volgende hoofdstuk worden behandeld.

7 Producten

Dit hoofdstuk behandeld de producten binnen het project. In het hoofdstuk “Implementatieplan en conclusie” zal besproken worden welke onderdelen zijn gerealiseerd.

7.1 Noodzakelijke producten

De noodzakelijke producten zijn verplicht voor het project. Dit zijn de minimale eisen die zijn gesteld.

Basis model

De visuele basis van het menu, zoals dat in het ontwerp is bepaald. Compleet met de tussentijdse animaties.

Hoofdmenu

Het hoofdmenu bevat alle keuze mogelijkheden die noodzakelijk zijn.
Het hoofdmenu dient de volgende onderdelen te bevatten;

- New Game: Keuze uit 3 moeilijkheidsgraden, verdere aanroep niet mogelijk
- Load & Save: Opslaan en laden van dummy waardes
- Inventory: Bekijken en gebruiken van de dummy inventaris
- Quit: Afsluiten van het menu

Documentatie

De documentatie beschrijft de opdracht, de voortgang en het resultaat. De documentatie is onderverdeeld in een aantal delen.

- Scriptie: Dit hoofddocument, behandeld alle onderdelen van het project
- PvA: Het plan van aanpak geeft aan wat de opdracht is, welke werkwijze er gehanteerd zal worden en wie er betrokken is bij het project
- Presentatie: Aan het eind van het project zal een presentatie worden gegeven om het eindresultaat te tonen en uitleg te geven bij het product

7.2 Eventuele producten

Wanneer er na het voltooien dan de verplichte producten voldoende tijd beschikbaar is kunnen eventueel de volgende onderdelen worden gemaakt.

Extra onderdelen hoofdmenu

Extra onderdelen van het hoofdmenu zijn;

- Options: Instellen van verschillende (video) instellingen.
- Credits: Een overzicht van bronnen, gebruikte software en geraadpleegde personen.
- Console: Met handige tool om met korte commando’s instellingen te veranderen.

Schaduw

Onderzoek doen naar het toepassen van schaduw in het project

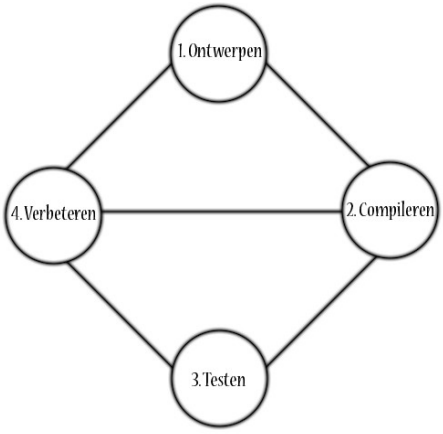
Geanimeerde modellen

De huidige statische modellen vervangen voor een geanimeerd model

8 Aanpak

Voorafgaand aan het project moet nagedacht worden over de uitgangspunten; wat gaat er gemaakt worden en hoe komen we tot een goed resultaat, welk basis ontwerp heeft de meeste mogelijkheden.
Een goed ontwerp voor het daadwerkelijke begin is zeer gewenst. Het schept duidelijkheid over wat er aan het eind van het project klaar moet zijn en welke stappen daarvoor nodig zijn. Uiteraard zullen tijdens het project problemen zich voordoen. Het goed uitdenken van het verloop kan dit wel beperken.

De vastgestelde werkwijze en routine voor het voltooien van het project is vastgesteld op een iteratiemode. Door het veelvuldig doorlopen van hetzelfde iteratiemodel worden onderdelen zorgvuldig gemaakt en is het mogelijk de voortgang goed bij te houden. Het gemaakte model is te zien in figuur 3.



Figuur 3

Elke iteratie zal beginnen met het maken van een ontwerp (1). Na het compileren(2) van de broncode zal een test(3) plaatsvinden. Na de test zullen eventuele verbeteringen(4) worden gemaakt, de broncode wederom gecompileerd en weer worden getest. Dit zal zich herhalen tot de werking naar verwachting functioneert.

Ontwerpfouten en handigheden die tijdens deze iteratie worden opgelost en verworven kunnen toegepast worden in de creatie van het volgende onderdeel. Dit gaat het maken van veel dezelfde fouten tegen en de snelheid zal toenemen.

9 Gemaakte keuzes

In dit hoofdstuk zullen de gemaakte keuzes besproken worden. Voornamelijk de grote beslissingen zullen aan de orde zijn, kleinere aanpassingen zijn meestal gemaakt om de efficiëntie van het programma ten goede te komen of een overzichtelijk geheel te creëren.

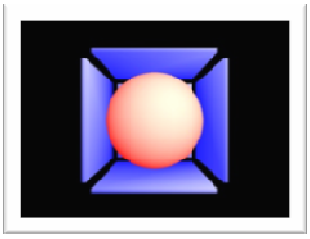
9.1 Ontwerp

9.1.1 Probleem

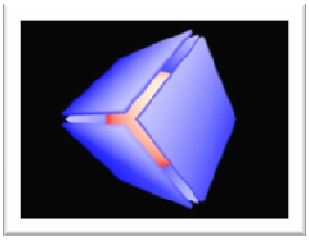
Het eerste ontwerp bestond uit één solide kubus. Het aangeven van de juiste positie was in dit ontwerp een groot probleem. Vaste coördinaten en rotatiewaarden om items op de juiste zijde te krijgen bleken de dynamiek van het project erg tegen te werken. Ook de mogelijke animatie zijn beperkt met een enkel object.

9.1.2 Oplossing

De oplossing is gekomen door de basis van een kubus in afzonderlijke delen te zien. Zes zijdes en een middenpunt. Het nieuwe ontwerp bestaat uit zeven onderdelen. Een bol in het midden, met daaraan vast de zes vlakken. Om een item op een vlak weer te geven is alleen een referentie nodig naar het vlak, de locatie van het vlak kan dan opgevraagd worden, evenals de rotatie. Er dient alleen nog aan gegeven te worden waar op het vlak het item moet komen. In vergelijking met de solide kubus is het plaatsen van een object nu eenvoudiger geworden, geen opgave van ruimtelijke plaatsing maar een kwestie van horizontale plaatsing en verticale plaatsing. Ook is het mogelijk de vlakken afzonderlijk te bewegen.



Figuur 4.1



Figuur 4.2

9.2 Meshes

9.2.1 Probleem

In het begin van het basismodelontwerp is gestart met het maken van meshes met 3D Studio Max. Hoewel het resultaat erg bevredigend is, visueel is veel mogelijk met het progrmma. Het grote nadeel is hierbij dat het maken van de meshes veel tijd kost. Tevens zal bij het vervolg van het project aanpassingen moeten worden gemaakt aan de zichtbare onderdelen, dit houd in dat een mesh weer aangepast moet worden waardoor veel tijd verloren zal gaan.

9.2.2 Oplossing

De engine zelf kent ook mogelijkheden om een mesh te creëren. Hoewel de afwerking van de door de engine gegenereerde meshes* minder zal zijn, afrondingen en andere visuele effecten zullen niet aanwezig zijn. Voordeel is dat de meshes snel aangepast kunnen worden. Hoogte, breedte, diepte, materiaal en andere variabelen zijn snel te veranderen per instantie. Het is dus niet noodzakelijk om voor elke breedte een geheel nieuw model te maken.

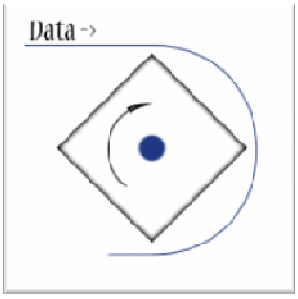
9.3 Carrousel

9.3.1 Probleem

Bepaalde menu's nemen een enkele zijde van het basis menu in beslag. Andere menu's zullen zoveel data bevatten dat dit niet op een zijde te plaatsen is. Het is dus zeker ook voor te stellen dat er zoveel data aanwezig is dat er onvoldoende zijdes voorhanden zijn om alles weer te geven.

9.3.2 Oplossing

De oplossing voor dit probleem is een vrij eenvoudig concept. Het basis model, in dit geval een kubus bevat naast de voor en achterkant, vier zijdes. Als we een as door de kubus denken, welke vast zit aan de voor- en achterkant van de kubus, is de kubus te draaien zodat we achtereenvolgens zijde één, twee, drie en vier zullen zien, waarna de cyclus weer opnieuw begint. Bij het verdelen van de data zal achtereenvolgens een zijde worden toegewezen. Wanneer alle data nu getoond zal worden staat alles door elkaar. Vandaar dat een carrousel is bedacht. We blijven de kubus draaien, en laten daarmee een teller lopen. Met deze teller kunnen we bijhouden welke data getoond moet worden. Door alleen deze data weer te geven is het mogelijk heel veel data te tonen op de vier beschikbare vlakken. Het figuur 5 geeft dit schematisch weer.



Figuur 5

9.4 XML

9.4.1 Probleem

Door XML bestanden te gebruiken voor de vaste data is een probleem ontstaan waar een goede oplossing voor gezocht moet worden. Omdat er verschillende types menu bestaan, elk met hun eigen plaatsing. Zo zullen bepaalde menu's een enkele zijde van het basismodel in beslag nemen, andere zullen meer vlakken in beslag nemen dan en voor handen zijn. Hoe geven we aan of het item op een volgend vlak moet komen. Hoe geven we aan hoe breed het item moet zijn, op welke plaats op het vlak moet het worden geplaatst?

9.4.2 Oplossing

Door met een XML bestand te werken kan verschillende data snel worden opgevraagd. Waardoor het aanmaken van een item door een methode kan worden afgehandeld. Deze methode kan dan zo gemaakt worden dat elk item met dezelfde methode kan worden gemaakt, in plaats van alle waarde uitschrijven. Het probleem van de vlakken is opgelost door met een zeer eenvoudige methode een teller op te hogen. Wanneer het nodig is dat het volgende item, en de onderliggende items op een volgend vlak moeten komen zal de waarde *Root* op "1" worden gezet. Een teller in de methode zal de waarde "1" optellen en weet zo dat het volgende vlak moet worden gebruikt. Wanneer het item op het zelfde vlak moet blijven is de waarde "0", bij de optelling met "0" blijft de waarde immers gelijk.
Andere waarde zoals de breedte zullen ook worden opgevraagd, opgeslagen en gebruikt bij het aanmaken van een item. Bij de beschrijving van de code zal dit ter spraken komen.

10 Implementatieplan en conclusie

10.1 Implementatie

Na voltooiing van dit project is de basis voor het menu gerealiseerd. De uiteindelijke bestemming voor dit menu is nog niet voltooid, waardoor het zeer waarschijnlijk is dat er in de toekomst meer veranderingen en aanpassingen zullen plaatsvinden. Bij de ontwikkeling van het computerspel waar het project in gebruikt zal gaan worden komen waarschijnlijk andere inzichten en mogelijkheden naar voren. Deze zullen worden doorgevoerd in het menu.
Echt sprake van een implementatie is er op dit moment nog niet.

10.2 Aanpassingen

Hoewel het menu dus nog aan veranderingen onderheven zal zijn in de toekomst, bevat het alle vereiste functies en is op een dusdanige manier opgebouwd dat aanpassingen en toevoegingen eenvoudig geïmplementeerd kunnen worden.
Er moet rekening worden gehouden met de verandering van bijvoorbeeld de inventaris. Niet zozeer het weergeven ervan maar eerder het beheer. Variabelen vanuit het spel zelf dienen nog op een manier te worden doorgegeven aan het menu. En visa versa. Het menu is nu gemaakt met een aantal dummy waarden, deze zijn niet zeer uitgebreid gemaakt maar hebben als doel de methode te testen. Later bij de implementatie worden deze dummy waarden vervangen voor een volwaardigere gegevensopslag. Bepaalde methoden zullen waarschijnlijk efficiënter gebruik kunnen maken van de aanwezige gegevens en beschikbare methodes van de engine* zelf. Hoewel de functionaliteit het zelfde zal blijven verdient een efficiënte werkwijze te worden geïmplementeerd, dit zeker met het oog op systeemeisen. Bijvoorbeeld de opslag methode. Zoals deze op dit moment aanwezig is exporteert de methode alle aanwezig bestanden. Dit moet veranderd worden in alleen de aangepaste bestanden.

10.3 Conclusie

Dit project is als try-out voor het vervolg project als geslaagd te beschouwen. De kennis van de omgeving is als goed te beschrijven, hoewel er nog veel te ontdekken valt bij het vervolg. Niet alle mogelijkheden zijn toegepast bij dit project omdat er simpelweg de noodzaak niet toe was. De kennis die is opgedaan is ruimschoots voldoende geweest om dit project tot het gewenste resultaat te brengen.

10.4 Resultaat

<< wat is er af? Wat niet en waarom?>>

11 Uitleg programma

In de volgende hoofdstukken zal de broncode worden toegelicht. Belangrijke methodes of delen ervan worden verklaard. De gehele broncode is als bijlage toegevoegd.

11.1 Controller

De controller is de kern van het programma. De controller houd bij welk menu actief is en geeft de input van de gebruiker door aan de desbetreffende logica.

11.1.1.1 Render

```
public void Render()
{
      TV.Clear();
      /// Render the Cube, always.
      Scene.RenderMeshList(7, basicMesh);
      Landscape.Render();
//////////////////////////////////////////////////////////////////
// Start 2D writing
      HUD.Action_Begin2D();
      HUDText.Action_BeginText(true);
      HUDText.NormalFont_DrawText(infoText, 5F,
      TV.GetViewport().GetHeight() -50, new TV_COLOR(1F, 1F, 1F,
      1F).GetIntColor());
      HUDText.Action_EndText();
      HUD.Action_End2D();
// Stop 2D writing
//////////////////////////////////////////////////////////////////
      if (runAnimation == false)
      {
            GetRenderMeshes();
      }
      TV.RenderToScreen();
}
```

Deze methode is verantwoordelijk voor het weergeven van de objecten op het scherm.
Te beginnen met het aangeven dat de buffer geleegd moet worden om een nieuw frame te tekenen. Hierna kan begonnen worden met het weergeven van objecten.
Scene.RenderMeshList heeft twee variabelen nodig. Het aantal objecten dat getekend moeten worden en een array van de index-waardes van deze objecten. Omdat het basis model vaste index waarden heeft (0 tot 6) is dit vast weergegeven.
Ook het landschap moet getekend worden. Omdat het landschap een bijzonder onderdeel is, niet gelijk aan een standaard mesh* bevat het landschap een eigen render methode.

Nu volgt het 2D overlay. Het tekenveld krijgt de opdracht te beginnen met het tekenen van 2D objecten. Direct hierna krijgt de 2DText de opdracht te beginnen met het weergeven van tekst. Gevolgd door het tekenen van de tekst, met enkele variabelen. De tekst, de horizontale locatie, de verticale locatie en de kleur van de tekst. Daarna zullen zowel de 2DText als het tekenveld de opdracht te stoppen met het aanmaken van objecten.

Wanneer er geen animatie plaats vind moeten ook de meshes* van de verschillende sub menu's worden opgevraagd. Deze methode zal meshes opvragen en met een RenderMeshList methode aan de Scene overdragen. Deze methode zal later behandeld worden.

Om alle onderdelen ook daadwerkelijk te tekenen zal de TV-engine alle objecten op het scherm moeten weergeven die de Scene klaar heeft staan. RenderToScreen voorziet in deze behoefte.

11.1.1.2 GetRenderMeshes

```
private void GetRenderMeshes()
{
      if (activeMenu.Equals("QUIT"))
      {
            Scene.RenderMeshList(Logic_Quit.GetRender().Length,
                                          Logic_Quit.GetRender());
      }
      else if (activeMenu.Equals("NEW"))
      {
            Scene.RenderMeshList(Logic_New.GetRender().Length,
                                          Logic_New.GetRender());
      }
      ....
      Scene.RenderMeshList(HJ_MainMenu.GetRender().Length,
                                          HJ_MainMenu.GetRender());
}
```

De waarde activeMenu zal worden aangepast wanneer een submenu actief word. Aan de hand van deze waarde kan de controller bepalen welk submenu actief en de gewenste objecten aanvragen. De lengte en de index waardes zullen aan de Scene worden gegeven. Als laatste zullen de lengte en de index waarden van het hoofdmenu worden opgevraagd. Deze zal altijd zichtbaar zijn wanneer er geen animatie plaats vind.

11.1.1.3 GetKeyInput

```
private void GetKeyInput()
{
      int key;
//////////////////////////////////////////////////////////////////
// non-repeating commands
//    (commands that should not continuously fire)
//////////////////////////////////////////////////////////////////
      Input.GetKeyBuffer(KEY_BUFFER, ref BUFFER_COUNT);
      for (int i = 0; i < BUFFER_COUNT; i++)
      {
            if (((TV_KEYDATA)KEY_BUFFER.GetValue(i)).Pressed != 0)
            // a key has been pressed!
            {
                  key = (int)((TV_KEYDATA)KEY_BUFFER.GetValue(i)).Key;
                  // get key int.
                  int ascii = Input.GetASCIIFromKey((CONST_TV_KEY)key);
                  //""""""""""""""""""""""""""""""""""""""""""""""""""
                  // Call the right logic. what menu has been activated?
                  if (activeMenu.Equals("MAIN"))
                  {
                        HJ_MainMenu.KeyPressed(key);
                  }
                  else if (activeMenu.Equals("NEW"))
                  {
                        Logic_New.KeyPressed(key, ascii);
                  }
                  ...
            }
      }
}
```

Bovenstaande is een deel van de invoer via het toetsenbord. De for-loop doorloopt de key-buffer, wanneer gedetecteerd word dat een toets is ingedrukt zal de waarde van de toets worden opgevraagd. Deze waarde is te gebruiken om later te bepalen welke toets het daadwerkelijk was. Een verschil is te zien in de aanroep van KeyPressed bij het hoofdmenu (HJ_MainMenu) en het nieuw spel submenu (Logic_New). Het verschil is gemaakt om het weergeven van letters mogelijk te maken. De ascii waarde is de letter die de toets representeert. Bij de behandeling van het nieuw spel submenu zal dit verder duidelijk worden.

11.1.1.4 Hoofdloop

```
while (bDoLoop)
{
      if (this.Focused)
      {
            Render();
            GetKeyInput();
      }
      else
      {
            System.Threading.Thread.Sleep(100);
            Input.ClearKeyBuffer();
      }
      Application.DoEvents();
}
TV.ReleaseAll();
TV = null;
this.Close();
```

De hoofdloop is een doorlopende cyclus. Wanneer het programma de focus heeft, oftewel het programma is geselecteerd in de taakbalk van het besturingssysteem. Zullen twee methodes worden aangeroepen. De methode Render om de actieve objecten weer te geven, en de methode GetKeyInput welke na gaat welke toetsen zijn ingedrukt.
Wanneer het programma de focus niet heeft hoeft het progamma niets weer te geven. Om de processor van de pc niet te veel te belasten zal het proces een wachttijd worden toegezegd. Wanneer het progamma de focus niet heeft zal er nog steeds worden bijgehouden welke toets is ingedrukt. Om dit te verhelpen en ongewenste handelingen in het programma tegen te gaan wanneer de focus weer word toegewezen zal de buffer worden geleegd.

DoEvents handelt de standaard optredende commando's af van het venster.
Wanneer de loop ten einde komt zal de engine alle objecten vrijgeven en worden uitgeschakeld waarna het hele programma ten einde zal komen.

11.2 HJ_XML_Reader

De XML lezer leest waardes uit een .xml bestand en slaat deze op een aantal arrays. Om een menu te creëren word een .xml bestand gebruikt om aan te geven welke variabelen deze bezit. De .xml bestanden dienen volgens een bepaalde standaard te worden gemaakt.

```
<Item
ID ="NEW" Root="0" Place="0.40" Ofset="0.0" Width="0.8" Name="New Game"
/>
```

De waarde ID is een waarde die voornamelijk gebruikt word voor controle mogelijkheden. Root geeft aan of het item op een volgende zijde van het basismodel moet worden getoond. Place geeft aan op welke hoogte, verticale locatie het item dient te komen, Ofset is de horizontale verplaatsing vanaf de standaard plaats. Width geeft aan hoe breed het item moet worden. 0.8 is de standaard waarde. Name, tot slot, geeft de tekst aan die op het item moet worden weergegeven.

De volgende arrays en variabelen worden gevuld in de XML lezer.

```
private string[] stringArray;
```

Bevat ID en Name

```
private float[] valueArray;
```

Bevat Ofset en Width

```
private float[] placeArray;
```

Bevat Place

```
private int[] rootArray;
```

Bevat de Root teller

```
private int arrayCounter;
```

Het aantal items in het bestand

```
private int departmentCounter;
```

Het aantal departementen, oftewel het aantal onderdelen. Een onderdeel zal per zijde worden weergegeven

11.3 HJ_MainMenu

Het Hoofdmenu bevat evenveel objecten als er submenu's zijn. Een object representeert een submenu en functioneert als een link. De objecten (meshes) worden in een array opgeslagen. De hoeveelheid objecten en de variabelen worden vanuit de XML-reader verkregen.

11.3.1.1 Build

```
public void Build()
{
      Controller.SetActiveMenu("MAIN");
      firstIndex = Controller.GetIndex();
      lastIndex = firstIndex;
      ReadXML();
      MeshArray = new HJ_View_MenuItem[arrayCounter];
      int k = 0;
      for (int i = 0; i < arrayCounter; i++)
      {
            MeshArray[i] = new HJ_View_MenuItem
                  (ref Scene, ref Root, stringArray[k + 1], placeArray[i],
                  valueArray[k], valueArray[k + 1], lastIndex,
                  stringArray[k]);
            MeshArray[i].Build();
            MeshArray[i].SetMaterial(mat1, mat2);
            k+=2; lastIndex+=2;
      }
      Controler.SetIndex(lastIndex);
      ActiveItem = MeshArray[0];
      ActiveItem.ToggleItem();
}
```

Om te beginnen word de controller op de hoogte gebracht van het actieve menu, waarna de eerst mogelijke indexwaarde opgevraagd zal worden. De laatste index word gelijk gemaakt aan de eerst mogelijke, waarna de laatste index per object met twee word opgehoogd. De waarde is twee omdat een object uit twee meshes bestaat, de balk die de achtergrond is, en de tekst op de balk. De waarde "k" word gebruikt voor het doorlopen van de array's met locatie waardes en tekst. Omdat beide array's meerdere waarden bevatten, bijvoorbeeld {lengte, breedte, lengte, breedte} waardoor niet de aanwezige iteratiewaarde "i" kan worden gebruikt.
Na de bouw-loop zal de index bij de controller worden vernieuwd zodat andere klassen bij de juiste indexwaarde zullen beginnen. Hierna zal het eerste object als actief worden gemarkeerd. In dit gaval het eerste object. Waarna de functie ToggleItem aangeroepen zal worden. Deze functie plaatst het object naar voren.

11.4 New Game

De Build methode van new game is grotendeels gelijk aan die van het hoofdmenu, er zal alleen een extra item worden gemaakt voor het invoeren van spelers naam. Daarom zal deze methode niet worden besproken.

11.4.1.1 Key input

De keypressed methode is ook grotendeels gelijk. Zoals eerder aangegeven is wel een extra variabele toegevoegd. Wanneer een moeilijkheidsgraad is gekozen dient de gebruiker een naam in te vullen. Deze naam zal worden gebruikt om het spel te herkennen in de opslag en laad methodes. Na het kiezen van een moeilijkheidsgraad zal een boolean waarden worden gezet, waardoor een apart deel van de input methode zal worden aangeroepen.

```
else if (selectedDiff == true)
{
      if (name.Equals("<Enter your name>")) { name = ""; i = 0; }
      if (key == (int)CONST_TV_KEY.TV_KEY_ESCAPE)
      {
            name = "<Select a diffilty>";
            MeshArray[MeshArray.Length - 1].Update(name);
            selectedDiff = false;
      }
      else if (key == (int)CONST_TV_KEY.TV_KEY_RETURN && i != 0)
      {
            CurrentGame = new HJ_File_Save(name, diff,
                              Controller.TakeScreenShot());
            selectedDiff = false;
            name = "<Select a diffilty>";
            MeshArray[MeshArray.Length - 1].Update(name);
            Controller.RotateCube("front");
            MeshArray[1].ResetSelected();
            MeshArray[2].ResetSelected();
            MeshArray[3].ResetSelected();
            Controller.SetActiveMenu("MAIN");
      }
      else if (key == (int)CONST_TV_KEY.TV_KEY_BACKSPACE && i > 0)
      {
            i--;
            nameInput[i] = (char)32;
            name = name.Substring(0, name.Length - 1);
            MeshArray[MeshArray.Length - 1].Update(name);

      }
      else if (ascii >= 65 && ascii <= 90 || ascii == 32
                                          || ascii >= 48 && ascii <= 57)
      {
            if (i < nameInput.Length)
            {
                  nameInput[i] = (char)ascii;
                  name += (char)nameInput[i];
                  i++;
            }
            MeshArray[MeshArray.Length - 1].Update(name);
      }
}
```

Beginnend met het verwijderen van de aangegeven informatie op de invoer balk, alleen als de tekst op de balk gelijk is aan de vorige informatie uiteraard. Als op de escape toets is gedrukt zal de tekst terug worden gezet naar de standaard, en zal terug worden gesprongen naar de selectie van de moeilijkheidsgraad. Wanneer de enter toets is ingedrukt en de waarde “i” niet nul is, wat betekend dat er invoer is geweest vanaf het toetsenbord. Zal een nieuw save-bestand worden aangemaakt met enkele standaard waardes, de naam, de moeilijkheidsgraad en een screenshot van het menu. Hierna zullen alle waarden worden teruggezet naar de originele toestand, de kubus zal worden gedraaid naar het hoofdmenu en het hoofdmenu zal worden geactiveerd. Met backspace kan een letter worden weg gehaald. Als geen van de drie bovenstaande toetsen is ingedrukt zal de ascii waarde die is mee gegeven worden omgezet in een karakter en toegevoegd aan een array. Op deze wijze kan de tekst op de balk worden aangepast.

11.5 Save Game / Load game

Load en save zijn grotendeels gelijk, beide gebruiken de opgeslagen data om objecten te tonen. Het verschil zit hem in het schrijven of lezen van de bestanden. Om dit te realiseren is een export en import methode gemaakt.

11.5.1.1 Import

```
public void ImportSave()
{
	saveCounter = 0;
	for (int i = 0; i < 20; i++)
	{
		try
		{
			Stream stream = File.Open("Save" + i + ".HJS",
				FileMode.Open);
			BinaryFormatter formatter = new BinaryFormatter();
			SaveFile[i] =
					(HJ_File_Save)formatter.Deserialize(stream);
			stream.Close();
			saveCounter++;
		}
		catch (Exception e)
		{
			Console.WriteLine("Error: import "+"Save" + i + ".HJS");
		}
	}
}
```

In totaal zijn er 20 plaatsen op het basismodel. 4 zijdes met 5 objecten. Deze loop zal 19 bestanden proberen te openen en te deserialiseren naar door het programma te lezen data. Deze zal worden opgeslagen in de array. Het aantal ingeladen bestanden word bijgehouden door de saveCounter.

11.5.1.2 Export

```
public void ExportSave()
{
	for (int i = 0; i < 20; i++)
	{
		try
		{
		Stream stream = File.Open("Save" + i + ".HJS",
					FileMode.Create);
		BinaryFormatter formatter = new BinaryFormatter();
		formatter.Serialize(stream, SaveFile[i]);
		stream.Close();
		}
		catch (Exception e)
		{
			Console.WriteLine("Error export");
		}
	}
}
```

Ook hier zijn de zelfde 20 plaatsen aanwezig. In dit geval zullen 20 bestanden worden gemaakt, of overschreven. De speldata zal worden geserialiseert naar een bestandsnaam met de plaats erin verwerkt. Save2.HJS bijvoorbeeld voor de 3de locatie.

12 Gebruiksaanwijzing

Hoe werkt het.. welke knoppen waarvoor?

13 Woordenlijst

Ontwikkelomgeving	De omgeving waarin het project tot stand is gekomen. Hiermee bedoelen we op welk platform (besturingssysteem) het project draait, welke software is gebruikt bij de ontwikkeling ervan
Engine / 3D Engine	De functionaliteit die het mogelijk maakt een driedimensionaal object in een tweedimensionale omgeving (beeldscherm) zichtbaar te maken.
Mesh / Meshes	Een mesh is de naam voor een 3D object. Elk onderdeel dat te zien is, is een mesh. Een kubus, bijvoorbeeld is een mesh.
Actor	Een dynamische mesh, een actor bezit een animatie die afgespeeld kan worden zonder dat de mesh aangepast moet worden.
Loop	Een herhalende functie. Zolang iets waar is, zal de loop door gaan met de herhaling.
Heightmap	Een zwart-wit plaatje waarmee een landschap kan worden gemaakt. Zwarte plekken gelden als laag, wit als zijnde hoog.

14 Bijlagen

Plan van aanpak

- titelblad, voorzien van titel, naam afstudeerder, examennummer, eerste examinator
- voorwoord
- managementsamenvatting (1 à 2 A-4's)
- inhoudsopgave
- inleiding met beschrijving van de organisatie van de opdrachtgever en de plaats van de
afstudeerder daarin
- de probleemstelling en de definitieve opdrachtomschrijving
- een beschrijving van de uitgangssituatie
- beschrijving van de op te leveren producten en/of het onderzoek
- een plan van aanpak voor de uit te voeren werkzaamheden
- een bespreking van mogelijke oplossingen met consequenties en motivering
- verdediging van de gekozen aanpak
- een beschrijving van de werkzaamheden en een analyse van de resultaten
- implementatieplan en zo mogelijk een beschrijving van de implementatie
- conclusies en aanbevelingen
- een evaluatie van de procesgang tijdens de afstudeerperiode
- geraadpleegde literatuur
- diverse bijlagen voor zover van belang