

ontwerpdocument

Individuele opdracht

Michelle Broens | S21M | 2017

Inhoudsopgave

[Inleiding 2](#_Toc477522283)

[Begrippen 3](#_Toc477522284)

[Architectuur 4](#_Toc477522285)

[Database ontwerp 5](#_Toc477522286)

[Het ontwerp 5](#_Toc477522287)

[Bijlage database ontwerp 6](#_Toc477522288)

[Tabellen 7](#_Toc477522289)

[Character 7](#_Toc477522290)

[npc 7](#_Toc477522291)

[Player 7](#_Toc477522292)

[Enemy 7](#_Toc477522293)

[Player\_SKill 8](#_Toc477522294)

[SKill 8](#_Toc477522295)

[Save 8](#_Toc477522296)

[World 8](#_Toc477522297)

[Level 9](#_Toc477522298)

[Tile 9](#_Toc477522299)

[Inventory 9](#_Toc477522300)

[Item 9](#_Toc477522301)

[Journal 9](#_Toc477522302)

[Quest 10](#_Toc477522303)

[Klassendiagram 11](#_Toc477522304)

[Klassendiagram bijlage 12](#_Toc477522305)

# Inleiding

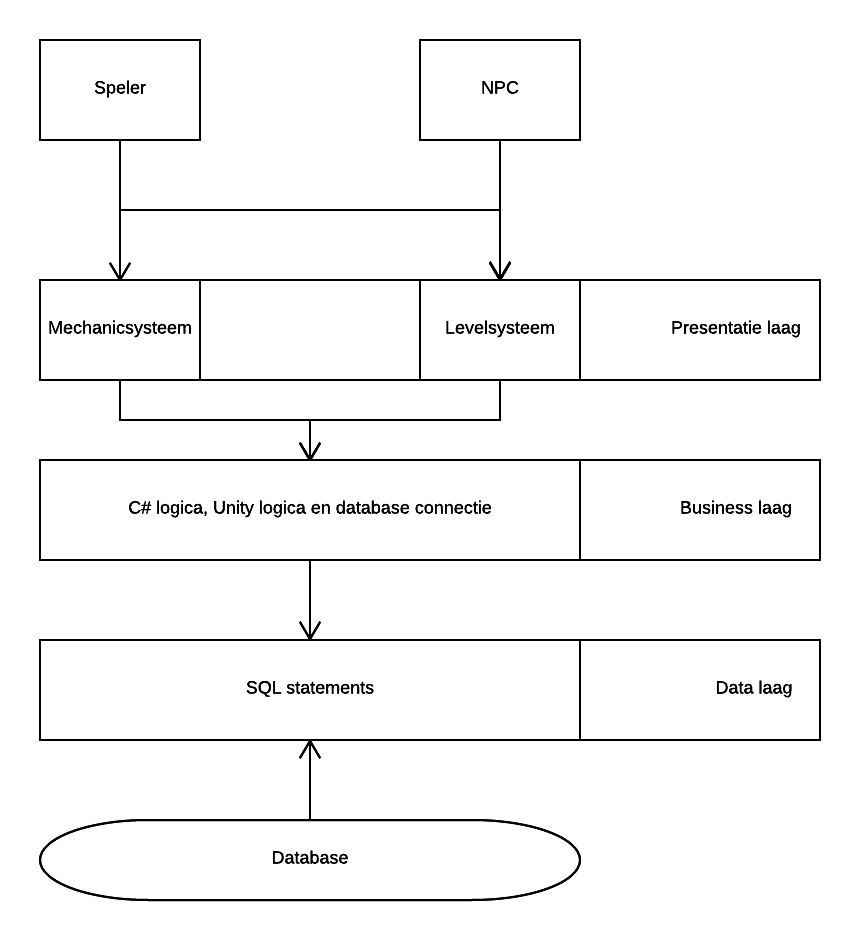
In dit document wordt er dieper ingegaan op de applicatie die in het document beschreven staat. Hier komen de meer gecompliceerde uitwerkingen die bijna meteen geïmplementeerd kunnen worden. Deze onderdelen zijn:

* Begrippen: dit is een lijst met begrippen hier staan woorden die de lezer misschien niet begrijpt of misschien niet helemaal begrijpt.
* Architectuur: dit geeft de structuur en communicatie van de applicatie weer in de vorm van een diagram
* Database ontwerp: hierin komt een diagram van de tabellen en attributen die in de database gezet worden, samen met een bijlage van dit diagram.
* Klassendiagram: dit geeft aan welke klassen, properties en methodes er in de applicatie gebruikt gaan worden.
* Infrastructuur: hier wordt in een diagram beschreven hoe de infrastructuur in elkaar zit, dus de server waar eventueel de database opstaat, de clients waarop de applicatie zou kunnen draaien.

# Begrippen

|  |  |
| --- | --- |
| **Begrip** | **Uitleg** |
| Property | In een property worden waardes opgeslagen om een privéveld buiten de klasse te kunnen gebruiken. |
| Methode | Een handeling die meerdere keren aangeroepen kan worden. |
| Infrastructuur | Een verzameling van voorzieningen die nodig is voor het transport van data. |
| Clients | Een client is een werkstation dus een pc of een laptop, die verbonden zit met de rest van de infrastructuur. |
| Query | Een query is een verzoek voor data/informatie van een tabel (of meerdere tabellen) aan de database. |
|  |  |
|  |  |

# Architectuur

De gebruiker logt in op het inlogscherm (presentatie laag) de gegevens worden doorgestuurd naar de buisiness laag. Deze gegevens worden door middel van een query (data laag) naar de database gestuurd.

# Database ontwerp

## C:\Users\miche\AppData\Local\Microsoft\Windows\INetCacheContent.Word\DBO.PNGHet ontwerp

## Bijlage database ontwerp

De **Character** tabel is een basis voor de andere entiteiten die van **Character**, dus de **NPC**, **Player** en de **Enemy** tabel. De **Enemy** heeft een veld genaamd **SpecialMove**, dit wil zeggen dat het een speciale **Enemy** is en deze een speciaal iets doet dat de andere niet doen, want normale **Enemies** kunnen alleen aanvallen. Een speciale move kan zijn dat de **Enemy** er extra schade bij krijgt.

De **NPC** heeft een speciaal veld voor de **Dialog**, dit is de tekst die de **NPC** gaat zeggen tegen de **Player**.

De **Player** heeft de meeste extra velden, ook wordt door **Journal\_ID** de speler aan een **Quest** gekoppeld. De tabel **Journal** is hier de koppeltabel tussen **Player** en **Quest**.

De **Quest** tabel heeft een veld genaamd **Description**, hier wordt beschreven wat er gedaan moet worden om de **Quest** af te ronden. Als de boolean **Progress** true is, is de **Quest** afgerond. De **Quest** wordt ook gekoppeld aan een **NPC**, want een **Player** kan alleen een **Quest** krijgen door met een **NPC** te praten en dan de **Quest** te accepteren.

De **Player** heeft ook **Skills**, deze **Skills** worden eerst aan de **Player** gekoppeld door de tabel **Player\_Skill**. Het veld **Description** legt uit wat de **Skill** doet, dus of het een vuurbal is of iets dergelijks. De **Enemy** heeft dezelfde koppeltabel tussen **Enemy** en **Skill**, maar deze heet **Enemy\_SKill**, dit heb ik gedaan omdat ik dit overzichtelijker vindt.

De **Player** heeft ook een **Inventory**, hier kunnen een aantal **Items** in, de **Type** bepaald of een item **Consumable** of **Equippable** is.

De **Save** tabel slaat op waar de **Player** is en wanneer de data opgeslagen is.

De **Character** tabel zit gekoppeld aan de **World** tabel, hier wordt opgeslagen welk **Character** op Welk **Level** zit, zo is het makkelijk om bepaalde **Enemies** te spawnen in een **Level**.

De **Level** heeft een **Type**, deze zegt of een **Tile** een muur is of een **Tile** waar de **Player** zo overheen kan lopen.

## Tabellen

### Character

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribuut** | **Type** | **Beschrijving** | **Verplicht** |
| **ID** | Integer |  | Ja |
| **Name** | Varchar(20) |  | Ja |
| **Level** | Integer |  |  |
| **HP** | Integer |  |  |
| **Mana** | Integer |  |  |

### npc

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribuut** | **Type** | **Beschrijving** | **Verplicht** |
| **ID** | Integer |  | Ja |
| **Dialog** | Varchar(20) |  |  |

### Player

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribuut** | **Type** | **Beschrijving** | **Verplicht** |
| **ID** | Integer |  | Ja |
| **Character\_ID** | Integer |  | Ja |
| **Journal\_ID** | Integer |  | Ja |
| **EXP** | Integer |  |  |

### Enemy

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribuut** | **Type** | **Beschrijving** | **Verplicht** |
| **ID** | Integer |  | Ja |
| **SpecialMove** | Boolean |  |  |

### Player\_SKill

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribuut** | **Type** | **Beschrijving** | **Verplicht** |
| **Player\_ID** | Integer |  | Ja |
| **Skill\_ID** | Integer |  | Ja |

### Enemy\_Skill

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribuut** | **Type** | **Beschrijving** | **Verplicht** |
| **Enemy\_ID** | Integer |  | Ja |
| **Skill\_ID** | Integer |  | Ja |

### SKill

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribuut** | **Type** | **Beschrijving** | **Verplicht** |
| **ID** | Integer |  | Ja |
| **Name** | Varchar(20) |  | Ja |
| **Damage** | Integer |  | Ja |
| **Description** | Varchar(80) |  |  |
| **ManaCost** | Integer |  | Ja |

### Save

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribuut** | **Type** | **Beschrijving** | **Verplicht** |
| **ID** | Integer |  | Ja |
| **Player\_ID** | Integer |  | Ja |
| **Date** | Date |  | Ja |
| **LocationX** | Double |  | Ja |
| **LocationY** | Double |  | Ja |

### 

### World

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribuut** | **Type** | **Beschrijving** | **Verplicht** |
| **ID** | Integer |  | Ja |
| **Character\_ID** | Integer |  | Ja |
| **Level\_ID** | Integer |  | Ja |

### Level

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribuut** | **Type** | **Beschrijving** | **Verplicht** |
| **ID** | Integer |  | Ja |
| **Type** | Varchar(5) |  | Ja |

### Tile

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribuut** | **Type** | **Beschrijving** | **Verplicht** |
| **ID** | Integer |  | Ja |

### Inventory

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribuut** | **Type** | **Beschrijving** | **Verplicht** |
| **ID** | Integer |  | Ja |
| **Item\_ID** | Integer |  | Ja |

### Item

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribuut** | **Type** | **Beschrijving** | **Verplicht** |
| **ID** | Integer |  | Ja |
| **Name** | Integer |  | Ja |
| **Type** | Varchar(10) |  | Ja |

### 

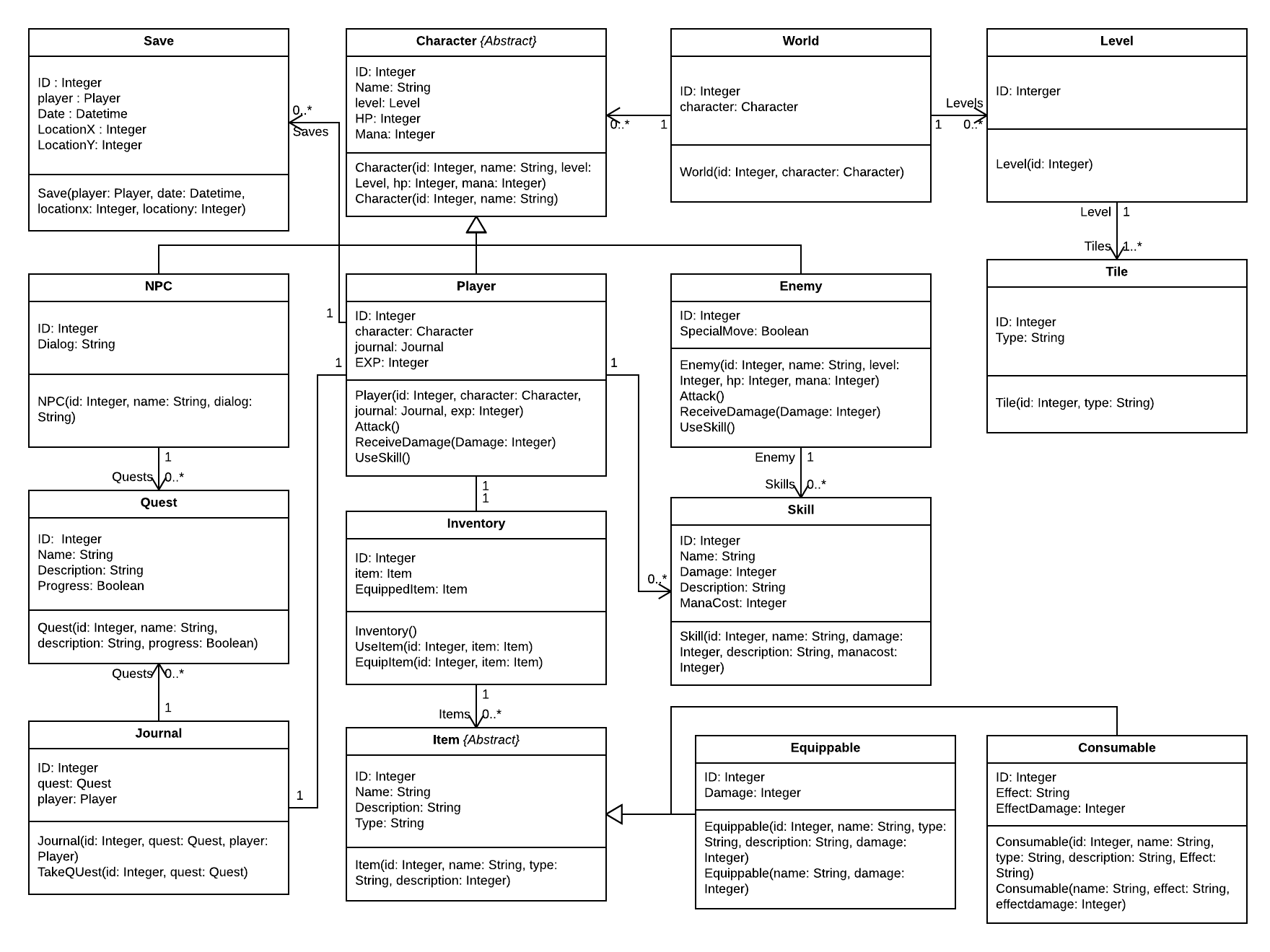
### Journal

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribuut** | **Type** | **Beschrijving** | **Verplicht** |
| **ID** | Integer |  | Ja |
| **Quest\_ID** | Integer |  |  |
| **Player\_ID** | Integer |  |  |

### Quest

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribuut** | **Type** | **Beschrijving** | **Verplicht** |
| **ID** | Integer |  | Ja |
| **Name** | Varchar(20) |  | Ja |
| **Description** | Varchar(80) |  | Ja |
| **Progress** | Boolean |  |  |

# Klassendiagram



## Klassendiagram bijlage

Ik ben ervan uit gegaan dat elke property een get, private set is.

De **Character** klasse is een abstracte klasse, er zijn drie klasse die hiervan af leiden: de **NPC**, de **Player** en de **Enemy**.   
De **NPC** is een karakter waar de speler **Quests** bij kan beginnen hier krijgt de speler dan een klein stukje **Dialog** te zien en daarna komt er een **Quest** tevoorschijn die de **NPC** geeft, als de speler deze accepteert komt de aangeroepen **Quest** in het **Journal** van de speler te staan. De rede dat een **NPC** ook overerft van **Character** is omdat ik het idee had dat er later ook **NPC** komen die meevechten met de **Player** of die je naar veiligheid moet brengen.

De **Player** zelf zit verbonden aan veel klasse. De **Player** heeft naast **Quests** ook een **Inventory** waar hij **Items** in op kan slaan. Er zijn twee verschillende soorten **Items**, dit zijn **Equippable**, denk aan wapens en andere uitrusting en **Consumable** denk aan potions die je HP herstellen.   
De **Damage** variabele bij de klasse **Equippable** is de schade die het wapen doet óf de defense die de uitrusting geeft.  
Bij **Consumable** is het **Effect** wat voor een effect dit item gaat hebben op de speler, dus een dat de speler voor een beperkte tijd meer schade doet, helen enz. Het **EffectDamage** is dan hoe veel het **Item** heelt. Een **Player** kan ook zijn progressie opslaan dit wordt gedaan in de Save klasse.

De **Enemy** heeft ook een **Skill**, maar een **Enemy** heeft niet altijd een **Skill** alleen de “belangrijke” **Enemies** hebben een **Skill**, dit zijn de bosses en mini-bosses.

Een **Character** staat op een **World**, er kunnen meerdere **Characters** op een **World**, want er zit altijd een **Player** op een **World**, maar ook meerdere **Enemies** en meerdere **NPC’s**. Een **World** heeft ook een **Level**, hier wordt de indeling van verschillende **levels** van **Enemies** opgeslagen en ook wat voor soort gebied het is, dit wordt gelijk getrokken aan scenes in Unity. Een **Level** is onderverdeeld in **Tiles**. In de **Tiles** wordt bij gehouden wat voor soort **Tile** het is, dus of het een muur is of een gebied waar de speler gewoon overheen mag lopen.