**MADD PROJECT: Crowfell**

**Significant Classes and Class Hierarchies**

**Significant Classes**

**Items**

|  |  |  |
| --- | --- | --- |
| **Access Modifier/Object Type/ Object Name** | **Inheritance Relations** | **Description** |
| Public Interface Item | N/A | Base object for an item. Holds basic methods for an object of type item. |
| Public Class Equipable | Implements Item | This object contains all common methods and variables that will belong to an equipable item such as a weapon or a piece of armour. It will also mean that the player may equip this item to their character. |
| Public Class Armour | Extends Equipable | This class contains all common methods and variables belonging to pieces of armour. This will be things such as Armour Rating durability etc. |
| Public Class Head | Extends Armour | This class describes a Helmet. This will be the lowest level of class. A helmet will be a new instance of this class. |
| Public Class Body | Extends Armour | This class describes a Chest Plate. This will be the lowest level of class. A Chest Plate item will be a new instance of this class. |
| Public Class Leg | Extends Armour | This class describes a Leg Plate. This will be the lowest level of class. A Set of greaves will be a new instance of this class. |
| Public Class Hand | Extends Armour | This class describes a pair of gloves. This will be the lowest level of class. A pair of gauntlets will be a new instance of this class. |
| Public Class Foot | Extends Armour | This class describes a pair of boots. This will be the lowest level of class. A pair of boots will be a new instance of this class. |
| Public Class Weapon | Extends Equipable | This class contains all common methods and variables belonging to weapons. This will be things such as min-max damage and damage type. |
| Public Class OneHanded | Extends Weapon | This class describes a one handed weapon. Weapons of this type will be held in the main hand unless specified that it may be held in the offhand (shields, dual wielding). Weapons of this type will create a new instance of this class. |
| Public Class TwoHanded | Extends Weapon | This class describes a two handed weapon. This type of weapon will be held in both hands and may not be dual wielded. Weapons of this type will create a new instance of this class. |
| Public Class Accessory | Extends Equipable | This class contains all common methods and variables belonging to accessories. This will be things such as attribute and skill boosts. |
| Public Class Ring | Extends Accessory | This class will describe a ring type item. This ring will have a list of attribute and skill boosts that aid the player at no cost (unless specified). |
| Public Class Necklace | Extends Accessory | This class will describe a Necklace type item. This necklace will have a list of attribute and skill boosts that aid the player at no cost (unless specified). |
| Public Class Consumable | Implements Item | This class creates the outline for a consumable item such as a potion or projectile. A consumable item has 1 use only and aids the player upon use such as healing or fired out of a weapon. |
| Public Class Potion | Extends Consumable | This class creates the outline for all potions. These potions will heal the player’s mana and health points, and will have a healing value stored within them. |
| Public Class Health | Extends Potion | This class describes a healing potion. It will be a single use item to heal the player.All healing potions will be an instance of this class and will be required to set a healing value. |
| Public Class Mana | Extends Potion | This class describes a mana potion. It will be a single use item to heal the player. All mana potions will be an instance of this class and will be required to set a healing value. |
| Public Class Projectile | Extends Consumable | This class creates the final outline for a projectile weapon. Projectiles will be consumed by the weapon that fires it upon firing. The damage type of ranged weapons will be determined by the projectiles fired by the weapon.  The damage dealt by projectile weapons will take the weapons and projectiles min-max damage into account to inflict the desired damage value. |
| Public Class Arrow | Extends Projectile | This class describes an arrow. All new arrows will be new instances of this class. The constructor for this class will require the following parameters: name, damage type, min-max damage. |
| Public Class Bolt | Extends Projectile | This class describes a bolt. All new bolts will be new instances of this class. The constructor for this class will require the following parameters: name, damage type, min-max damage. |
| Public Class Bullet | Extends Projectile | This class describes a bullet. All new bullets will be new instances of this class. The constructor for this class will require the following parameters: name, damage type, min-max damage. |

**Entities:**

|  |  |  |
| --- | --- | --- |
| **Access Modifier/Object Type/ Object Name** | **Inheritance Relations** | **Description** |
| Public Interface Entity | N/A | Base object for an Entity. Holds all basic common methods shared by Entities within the game. |
| Public Class ActiveEntity | Implements Entity | Base object for an active entity. An active entity is defined by an entity that can move. This contains the basic structure shared by all moving entities. |
| Public Class Monster | Extends ActiveEntity | This class outlines an entity that has its own AI. This cannot be physically controlled by the player. AI methods will be added in here. |
| Public Class Friendly |  | This class outlines a friendly monster. A friendly monster does not harm the player and will aid in the destruction of evil. AI for that type of monster will be implemented here. Also the monster will need a link to its owner in the case of a multiplayer game. This entity will follow the player around and when a monster is near will be drawn to attacking it. All new friendly monsters will either need to be a new instance of this class or will need to extend this class. |
| Public Class Evil |  | This class outlines an evil monster. And evil monster will attack the player on sight. It will have a line of sight and earshot radius to determine whether it has detected the player. Upon detection the monster will move directly to the player. It will also have to fight the friendly monsters and will have detection for them too.  All new evil monsters will either need to be a new instance of this class or will need to extend this class. |
| Public Class Player |  | This class outlines the player’s entity. This will have methods to control and update the player. It will also have links to its equipped items and will have an inventory list. |
| Public Class InactiveEntity |  | This object outlines the in-active entities. These will be entities that do not move but still can be interacted with. |
| Public Class Merchant |  | This class outlines the basic structure for a merchant. These can be interacted with and will have a list of current items they have for sale.  All new merchants will need to be a new instance of this class or extend this class. |

**Map**

|  |  |  |
| --- | --- | --- |
| **Access Modifier/Object Type/ Object Name** | **Inheritance Relations** | **Description** |
| Public Class Map | N/A | This class will hold the details of a dungeon map. This will be things such as the tile map flooring (2D Array), height and width (as number of tiles), and the methods to draw and generate the map. Different map types will be stored as an enum data type and one will be chosen at random. This will influence how a map will be generated. |
| Public Class MapRoom | N/A | This class will hold the details of a room within the map. This will include things such as width and height (as number of tiles), location within tile map (location of the top left corner tile of the room), room type etc.  Will contain the methods to generate the room at that location. |
| Public Class MapTile | N/A | This class will hold the details of a single tile within the tile map. This will be things such as the mesh size, the feature drawn specific to the tile (tile features include water, lava, walls, barrels etc.). |
| Public Class MapFeature | N/A | This class will be the outline for all map features such as walls and large rocks. Small features will fit on a single tile and will be drawn on a specific tile. However larger features will just be given a starting map tile (location) and will be drawn with the given location at the centre of the feature. |

**Hierarchies**

**Items Hierarchy**

Interface Item

- Class Equipable implements Item

- Class Armour extends Equipable

- Class Head extends Armour

- Class Body extends Armour

- Class Leg extends Armour

- Class Hand extends Armour

- Class Foot extends Armour

- Class Weapon extends Equipable

- Class OneHanded extends Weapon

- Class TwoHanded extends Weapon

- Class Accessory extends Equipable

- Class Ring extends Accessory

- Class Necklace extends Accessory

- Class Consumable implements Item

- Class Potion extends Consumable

- Class Health extends Potion

- Class Mana extends Potion

- Class Projectile extends Consumable

- Class Arrow extends Projectile

- Class Bolt extends Projectile

- Class Bullet extends Projectile

**Entities Hierarchy**

Interface Entity

- Class ActiveEntity implements Entity

- Class Monster extends Active

- Class Friendly extends Monster

- Class Evil extends Monster

- Class Goblin extends Evil

- Class Undead extends Evil

- Class Human extends Evil

- Class Troll extends Evil

- Class Demon extends Evil

- Class Weapon extends Equipable

- Class OneHanded extends Weapon

- Class TwoHanded extends Weapon

- Class Accessory extends Equipable

- Class Ring extends Accessory

- Class Necklace extends Accessory

- Class InactiveEntity implements Entity

- Class Merchant extends InactiveEntity

- Class Smithy extends Merchant

- Class Alchemist extends Merchant