FIT2099 Assignment 3(Updates)

Class Responsibilities

(Braaaains)

Group Name	:	Highroll
Group Members	:	Tan Weihan
		Wong Jun Kang
Tutor	:	Dr Jasbir Kaur Dhaliwal
Tutorial Datetime	:	Tuesday, 4:00pm-6:00pm
Assignment	:	Assignment 3

FIT2099 Assignment 1 Class Responsibilities

Zombie

Zombie class is a child class of Actor class. Zombies can perform behaviours such as SpeechBehaviour, AttackBehaviour, PickUpItemBehaviour, HuntBehaviour and WanderBehaviour.

- This class will contain 2 int type instance variables that specify the number of arms and legs on a zombie.
- A zombie will have 2 arms and legs upon creation.
- Zombies can not lose more than 2 arms and 2 legs.
- A zombie can drop more than 1 limb in 1 attack, the amount of limb to drop will be calculated. A method will be created to calculate the drop rate of the Zombie's limb.
- Zombies will have a 25% chance to drop 1 or more limbs.
- To drop a limb, the Zombie object will call a method to drop limbs in the Zombie class. The method will decrement the number of legs or the number of arms of the Zombie Object.
- When a Zombie object loses its limb(s) a ZombieLimb CraftableItem object will be created and added onto the ground where the Zombie is located.
- When a Zombie object loses its arm(s), it will be half as likely to punch.
- A method calculateLimbDropRate is used to calculate the drop rate of the zombie's limb.
- When a Zombie object loses a leg, it can only invoke behaviour with MoveActorActions such as HuntBehaviour() or WanderBehaviour() every other turn (or once every 2nd turn).
- When a Zombie object loses both legs, it can not invoke any behaviours that involve MoveActorActions such as HuntBehaviour() or WanderBehaviour(). Only behaviours like AttackBehaviour() or DoNothingAction() can be invoked.
- If the zombie is currently holding a weapon item, if it is there will be a 50% chance for the zombie to drop the weapon.
- When a Zombie loses its leg(s) or arm(s), a ZombieLimb will be created and added to the current location of the Zombie.
- A new intrinsic weapon named "bite" will be created.

- Zombies will also bite, a bite deals more damage than a punch attack and will heal the zombie for 5hp.

- There is a 50% chance set for a Bite attack, otherwise, a punch attack will be triggered.

- Zombies will also have a 10% chance to perform any zombie-like speech.

- The zombie will take a turn to perform a speech, by invoking a speech behaviour.

- Zombies can pick up a weapon if there is one on its location.

- A zombie can only possess one WeaponItem at a time. Once a zombie picks up a WeaponItem, it cannot pick up a second one, unless the zombie drops its weapon.

- A method punchOrBite is implemented to calculate and decide which intrinsic weapon the zombie will use.

SpeechAction

Allows actors to make a speech.

- A turn will be taken/ spent to perform a speechAction.

SpeechBehaviour

SpeechBehaviour class implements Behaviour interface.

- SpeechBehaviour allows actors to perform SpeechAction.

CraftableItem

CraftableItem class extends PortableItem class.

CraftableItem class has an additional parameter “newItem” to take in an Item that indicates what the current item can be crafted into.

- CraftableItem are portable items that can be crafted into other items.

- CraftableItem contains an additional allowableAction that allows the actor having the item to perform a CraftAction.

CraftAction

CraftAction class extends Action class

CraftAction takes in 2 parameters, “oldItem” and “newItem”

- CraftAction when given 2 items “oldItem” and “newItem”, craftAction replaces the “oldItem” with the “newItem” in the inventory.
    - CraftAction first removes the “oldItem” from the inventory of the user.
    - CraftAction then adds the “newItem” into the inventory of the user.

ZombieLimb
ZombieLimb class extends CraftableItem class and implements Weapon interface.

ZombieLimb is a CraftableItem that can be crafted into a ZombieClub and Zombie Mace.

- ZombieLimbs can be crafted into upgraded items.

- CraftableItem such as the ZombieLimbs will have 2 allowable actions.
    - Crafted into upgraded item
    - Attack with Zombie's leg/ hand

- When craftAction is chosen, the craftable item in the player's inventory will be replaced with it's upgraded item.

ZombieClub
ZombieClub class extends WeaponItem class.

ZombieClub is a weapon item that deals more damage than ZombieLimb.

ZombieMace
ZombieMace class extends WeaponItem class.

ZombieMace is a weapon item that deals more damage than ZombieClub.

HumanCorpse
HumanCorpse class extends PortableItem class

HumanCorpse will "rise from dead" to become a Zombie after a certain amount of playturns. HumanCorpse will be added onto the Human's location(performed in AttackAction) when a Human dies.

- HumanCorpse class overrides the method tick() from the item class.

- HumanCorpse class implements a countTurn counter that will increment by 1 whenever a turn passes.

- A constant is added in the class to indicate the turn needed for a HumanCorpse to be transformed into a Zombie object.

- A second counter is created and is initialised to the constant value.

- The second instance will increment when an Actor was detected to be standing on the HumanCorpse object during the turn when a Zombie is supposed to be created. This delays the revive turn and ensures no 2 Zombie Objects will exist in the same location.

- A HumanCorpse is a PortableItem. It can be picked up by PickUpItemAction. When picked up, the reviveTurn will be incremented signifying a delay of “rise from death” until it is dropped back to the ground.

- When the first instance is equal to the second instance counter, a Zombie object will be added to the location of the corpse and the HumanCorpse object will be removed from the map.

Farmer

Farmer class extends Human class.

Farmers are humans with additional functionalities and behaviours.

- Farmers are given 3 additional behaviours including, SowBehaviour, FertiliseBehaviour and HarvestBehaviour.

- The priority of the behaviours are SowBehaviour, HarvestBehaviour, followed by FertiliseBehaviour.

- Farmers will plant crops, fertilise crops and harvest crops

- When planting crops, Dirt objects will turn into Crop objects.

- When fertilise crops, crop age will increase by 10

- When harvesting crop, crop will turn into dirt, and a food object is created

Crop

Crop class extends Ground class.

Crop objects are Ground objects that are given 2 additional Capabilities including “*UNRIPE”* and *“RIPE”* and some additional methods and functionalities.

- Crop objects have an instance variable that indicates the current age of the crop.

- Crop class overrides the method tick in the item class, each time when a turn passes, the age of the crop will be incremented by 1.

- Crop objects can only be planted on Dirt type objects that have the “*SOWABLE”* capability.

- Crop objects will take 20 turns to ripen, while not ripe, Crop objects will have a UNRIPE capability

- Once ripened, a new capability *RIPE* will be added into the Capability set of the crop object, and its UNRIPE capability removed.

- Crop objects with UNRIPE capability can be fertilised, reducing time needed to ripen by 10 turns.

- When ripened, crops can be harvested into food by farmers or the player via HarvestAction.

- When harvested, the location of the crop will turn back into dirt.

Dirt

An additional Capability "*SOWABLE"* will be added to the Dirt object.

SowAction

SowAction class extends Action class.

SowAction allows farmers to plant crops. Replaces dirt object with a new crop object

SowBehaviour

SowBehaviour class implements Behaviour interface.

SowBehaviour applies SowAction onto dirt objects.

- SowBehaviour ensures only 33% of the time a SowAction will be invoked, otherwise, null will be returned.

- SowBahaviour checks whether the ground object has the Capability of "*SOWABLE*" before applying SowAction to the ground object.

- Since, only dirt objects are given the Capability of "*SOWABLE".* SowBehaviour() can ensure that only Dirt() can be set into a Crop() type object.

FertiliseAction

FertiliseAction class extends Action class.

FertiliseAction allows farmers to fertilise crops. Increases the age of crop by 10.

- FertiliseAction will invoke a method fertilise() from the Crop class.

- The method fertilise() will increase the age of the crop by a specified constant (GROW_RATE) of 10.

FertiliseBehaviour

FertiliseBehaviour class implements Behaviour interface.

FertiliseBehaviour applies FertiliseAction onto crops.

- FertiliseBehaviour checks whether the Crop object has the Capability "*UNRIPE"* before applying FertiliseAction to the Crop object.

- Since, only Crop objects are given the Capability of "*UNRIPE".* FertiliseBehaviour can ensure that only Crop objects can be fertilised.

HarvestAction
HarvestAction class extends Action class.

HarvestAction allows farmers and players to harvest crops. Replaces crop object with a new dirt object. Place a food object at the location of the new dirt object. If a player calls HarvestAction, food will be placed in their inventory. Only works on ripe crops (age >= 20).

- Since, only Crop objects are given the Capability of "*RIPE".* FertiliseBehaviour() can ensure that only Dirt() can be set into a Crop() type object.

- A Crop object will only obtain a new Capability "*RIPE"* when the age of the crop >= 20, hence, HarvestAction will not be invoked on unripe Crop objects.

HarvestBehaviour
HarvestBehaviour class implements Behaviour interface.

HarvestBehaviour applies HarvestAction onto crops.

- HarvestBehaviour checks whether the Crop object has the Capability "*RIPE"* before applying HarvestAction to the Crop object.

- Since, only Crop objects with age> 20 are given the Capability of "*RIPE".* HarvestBehaviour can ensure that only ripe Crop objects can be fertilised.

- HarvestBehaviour ensures farmers can harvest adjacent crops or crop directly below them.

EatAction
EatAction allows humans to eat harvested food. Heals them by a certain amount when done.

EatBehaviour
EatBehaviour generates an EatAction when there is food in the Actor's inventory.

Food
Food class extends PortableItem class.

Food is created when a ripe crop (age >=20) is harvested via HarvestAction. Heals a certain amount when used in EatAction.

PickUpItemBehaviour

PickUpItemBehaviour class implements Behaviour interface.

Allows actors (Zombie & Human) to pick up items.

- Zombies objects can pick up Weapon with this behaviour
- Human objects can pick up Food with this behaviour

- Restricts Zombie from picking up a Weapon if:
    - The zombie has no arm.
    - The zombie already has a weapon on its hand.

AttackAction

- If the weapon returned by getWeapon() method is bite, AttackAction will use a random number generator to determine if it hits the target.

- A HumanCorpse object will be created when isConsious method in the AttackAction detects Human's hitpoint becomes smaller or equal to 0.

- A method is implemented in AttackAction class to decide whether an attack will be landed successfully based on the accuracy of each attack.

Assignment 3 (Additional Functionalities)

RunAwayBehaviour

RunAwayBehaviour class implements Behaviour class.
This class returns a MoveAction that moves a current actor away from actors with ZombieCapability.UNDEAD.

- Checks for all tiles around the actor and returns a score for each surrounding tile.
- Score represents zombie count, higher zombie count higher score
- Actor will move to tile with least score
- If no Undead nearby return null and execute other behaviour.
- Used by Human type Actors.

RangedWeapon

RangedWeapon class extends WeaponItem class
The base class for ranged weapons like SniperRifle and Shotgun.

- All RangedWeapon are given ItemCapability.RANGED.
- Contains methods like addAmmo(), minusAmmo() and getAmmoCount() to keep track of current ammoCount in the RangedWeapon.
- Contains getter for damage and accuracy.
- getAccuracy allows actors to get the accuracy of current RangedWeapon.

Shotgun

Shotgun class extends RangedWeapon class.
Class for creating a Shotgun object.

- Damage: 30, Accuracy: 75
- Has ItemCapability.SHOTGUN
- Initial ammoCount is set to 3
- Given an additional allowable action ShotgunMenu to access if Shotgun is inside the player's inventory and has an ammoCount greater than 0.
- ShotgunMenu, will allow the user to select a direction to fire the Shotgun.
- Total of 8 available directions: North, North-East, East, South-East, South, South-West, West, North-West.
- Damage any actor except the player himself/ herself within the area of effect.

ShotgunAction

ShotgunAction class extends Action class.The main logic of Shotgun.
Applies AttackActions to actors within the range of the selected direction of the Shotgun.

- Decrease ammoCount of Shotgun by 1, whenever ShotgunAction is called.
- Display a message indicating the success of failure(misses) of the shoots along with the menuDescription.

ShotgunMenu

ShotgunMenu class extends Action class.
A menu that allows the user to select a direction to fire the Shotgun.

- Players can only fire the Shotgun if the exit exists, meaning that player can't fire his/her Shotgun towards the boundary.
- ShotgunActions of all available directions (exits) are added into a menu for the player to select from.

SniperRifle

SniperRifle class extends RangedWeapon class.
Class for creating a SniperRifle object.

- Default Damage: 45, Default Accuracy: 75
- Has ItemCapability.SNIPERRIFLE
- Initial ammoCount set to 2
- Given an additional allowable action SniperRifleMenu to access if SniperRifle is inside the player's inventory and has an ammoCount greater than 0.
- SniperRifleMenu will allow the player to select a target to perform SnipeAction at.
- Target can only be actors with ZombieCapability.UNDEAD.
- The method getAccuracy is overridden from the RangedWeapon to adjust the accuracy of the weapon depending on the aimCount.
- The method getDamage, will return a damage depending on aimCount.
- When aimCount = 0, Damage: DEFAULT_DAMAGE (45)
  Accuracy: DEFAULT_ACCURACY (75)

- When aimCount = 1, Damage: DEFAULT_DAMAGE * 2 (90)
  Accuracy: AIM_ACCURACY_1 (90)

- When aimCount = 2, Damage: INSTAKILL (Integer.MAX_VALUE)
  Accuracy: AIM_ACCURACY_2 (100)

SnipeAction

SnipeAction extends Action class.

- SnipeAction takes in a target parameter set target onto the SniperRifle object.
- Return a submenu with 2 available actions, AttackAction to fire SniperRifle or AimAction to continue to aim.
- The player will be given an AimAction only if current aimCount < 2.

SniperRifleMenu

SniperMenu class extends Action class.

If target is null for SniperRifle, loop through the entire map, and return a list of menu of actors that can be targeted by the SniperRifle, else, return SnipeAction and the player can choose between aiming or sniping.

- Target must have ZombieCapability.UNDEAD

Ammo

Ammo class extends PortableItem class.

The base class for all ammunition of RangedWeapon.

This class allows creation of ammunition for RangedWeapon(guns) .

- An abstract class.
- This item will allow for ReloadAction when the specified Weapon for the Ammo is available in the player's inventory.
- Ammo must be inside the player's inventory for ReloadAction to appear.

ShotgunAmmo

ShotgunAmmo class extends Ammo class.

The class for ammunition of Shotgun

- Each ShotgunAmmo object provides the player with a default of 3 ammos.

SniperRifleAmmo

SniperRifleAmmo class extends Ammo class.

The class for ammunition of SniperRifleAmmo.

- Each SniperRifleAmmo object provides the player with a default of 2 ammos.

AimAction

AimAction extends Action class.

This class allows RangedWeapon (SniperRifle)to perform aiming action.

- Used only by SniperRifle Weapon Item.
- SniperRifle aims with AimAction
- AimAction increments the aimCounter of the selected Weapon.

ReloadAction

ReloadAction class extends Action class.

This class allows for actions that reload ammo for RangedWeapon.

- Called addAmmo() for the specified RangedWeapon to increment ammoCount for the chosen weapon.

MamboTotem

MamboTotem class extends Item class.
This class determines the spawning of Mambo Marie, along with storing Mambo Marie's condition (eg; killed).

- Has 2 Boolean values, killed and spawned. Killed checks if Mambo Marie has been killed by the player, while spawned checks if Mambo Marie is currently on the map.
- Method tick() controls the spawn of Mambo Marie by checking if she is not killed and not spawned, along with using a random number generator.
- Method marieSpawn() spawns Mambo Marie.
- Method resetSpawn() sets spawned back to false. This means that she is no longer on the map and allows MamboTotem to spawn her again.
- Method setKilled() sets killed to true. This means that Mambo Marie is killed.
- Method getKilled() returns Boolean killed. Used to check if Mambo Marie is killed

MamboMarie

MamboMarie class extends ZombieActor class.
MamboMarie is a voodoo priestess that summons zombies.

- Method playTurn determines her actions
    - If 30 turns have passed since her appearance, she will leave the map.
    - Every 10 turns, she will spawn 5 zombies via SpawnAction

SpawnAction

SpawnAction class extends Action class.
Spawn Action spawns 5 zombies at random locations.

Vehicle

Vehicle class extends Item class.
Vehicle allows the player to travel between maps.

- Travelling between maps done via DriveAction

DriveAction

DriveAction extends Action class.
DriveAction transports the player to another map.

EndableWorld

EndableWorld extends World.
EndableWorld adds endings to game.

- Checks for 2 things, a Boolean called end and player's existence in game.
- If end is false and player does not exist, it means player quits game
- Else, it'll determine win or lose by counting the humans and zombies present in maps of the game.

QuitAction

QuitAction extends Action.
QuitAction allows players to quit the game.

Enums

ZombieCapability

- An additional enum PLAYER is added into ZombieCapability enum class.
- An additional enum MAMBO is added into ZombieCapability enum class.

FarmingCapability

The enum class is used to assign specific ground type (dirt) or crop with a status.
The class contains 3 enums.

- SOWABLE- implies that aSowAction can be performed to the Ground(Dirt).
- UNRIPE- implies that a FertiliseAction can be performed to the Crop.
- RIPE- implies that a HarvestAction can be performed to the Crop.

ItemCapability

The enum class is used to assign an Item object with a “type”.

- MELEEWEAPON- implies that the current Item is a melee weapon instance.
- RANGEDWEAPON- implies that the current Item is a ranged weapon instance.
- SHOTGUN- implies that the current Item is a Shotgun instance.
- SNIPERRIFLE- implies that the current Item is a SniperRifle instance.
- FOOD- implies that the current Item is a Food instance.
- VEHICLE- implies that the current Item is a Vehicle instance.
- TOTEM- implies that the current Item is a MamboTotem instance.
- INVENTORY- implies that the current Item is currently inside the inventory of the player.