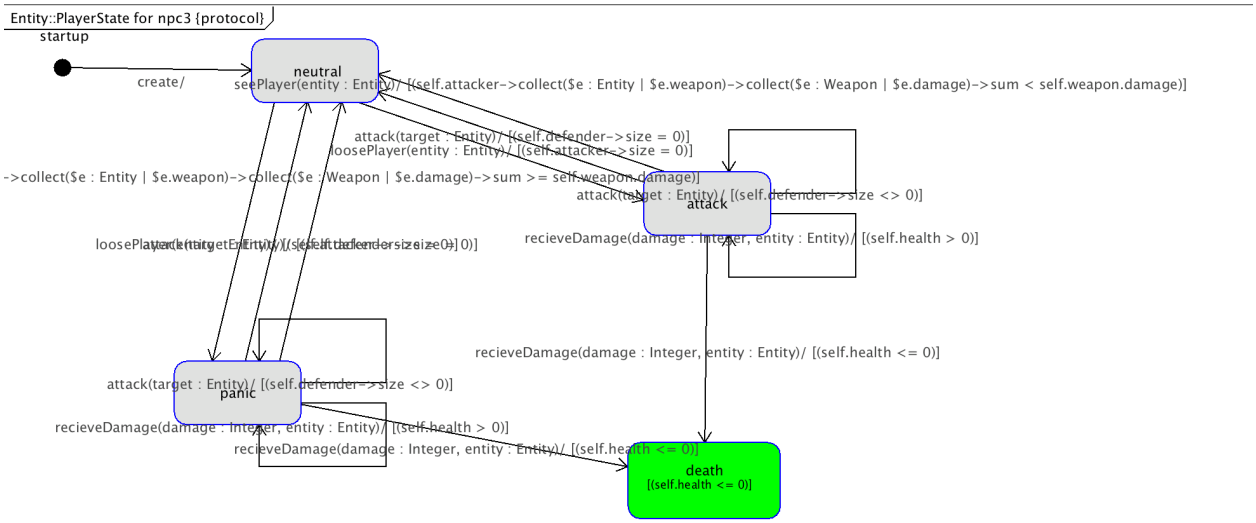
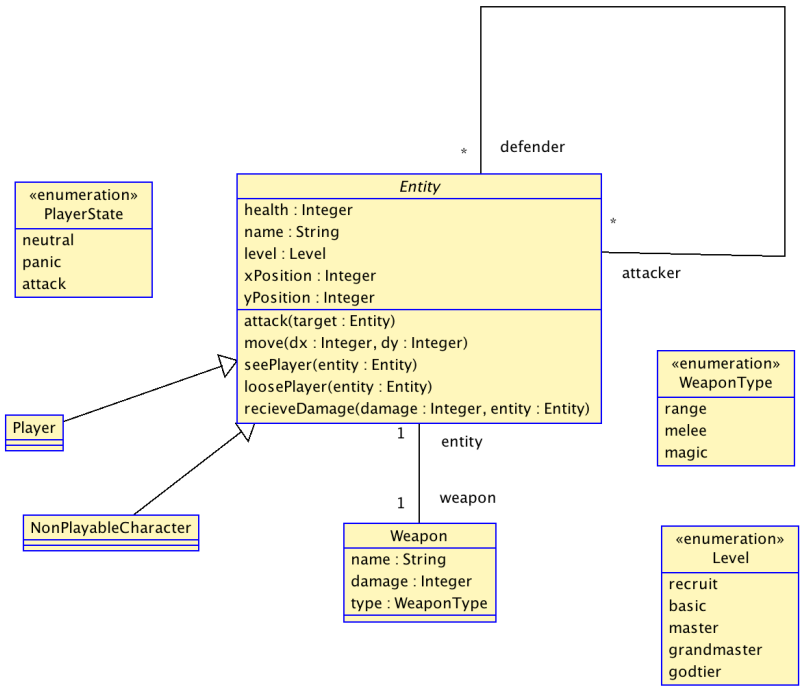
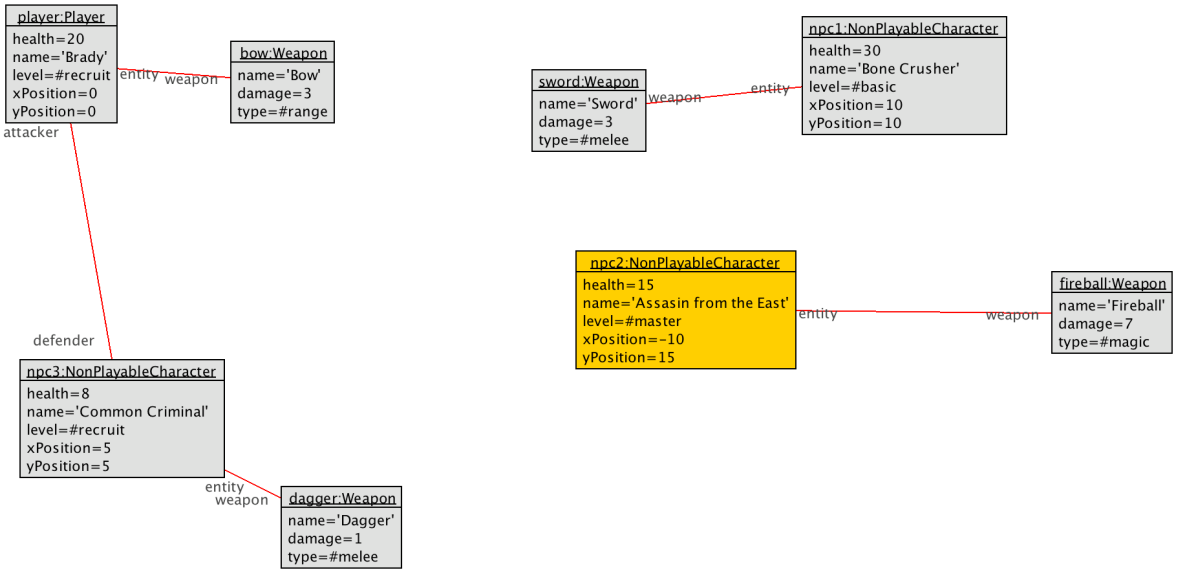
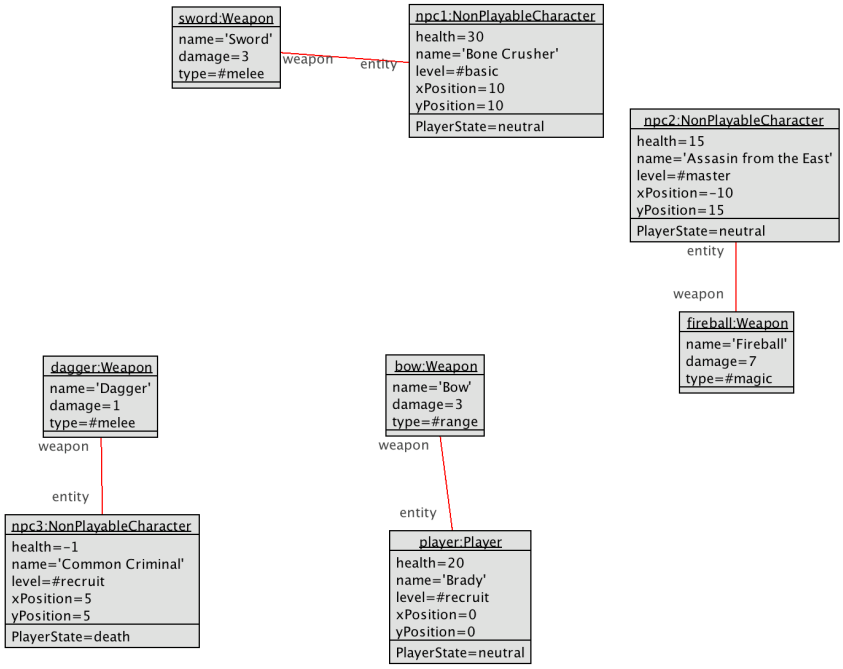

Entity::PlayerState for npc3 {protocol}
startup
neutral
create/
seePlayer(entity : Entity)/ [(self.attacker->collect($e : Entity | $e.weapon)->collect($e : Weapon | $e.damage)->sum < self.weapon.damage)]
attack(target : Entity)/ [(self.defender->size = 0)]
loosePlayer(entity : Entity)/ [(self.attacker->size = 0)]
->collect($e : Entity | $e.weapon)->collect($e : Weapon | $e.damage)->sum >= self.weapon.damage)]
attack(target : Entity)/ [(self.defender->size <> 0)]
attack
recieveDamage(damage : Integer, entity : Entity)/ [(self.health > 0)]
loosePlayer(entity : Entity)/ [(self.defender->size = 0)]
recieveDamage(damage : Integer, entity : Entity)/ [(self.health <= 0)]
attack(target : Entity)/ [(self.defender->size <> 0)]
panic
recieveDamage(damage : Integer, entity : Entity)/ [(self.health > 0)]
recieveDamage(damage : Integer, entity : Entity)/ [(self.health <= 0)]
death
[(self.health <= 0)]

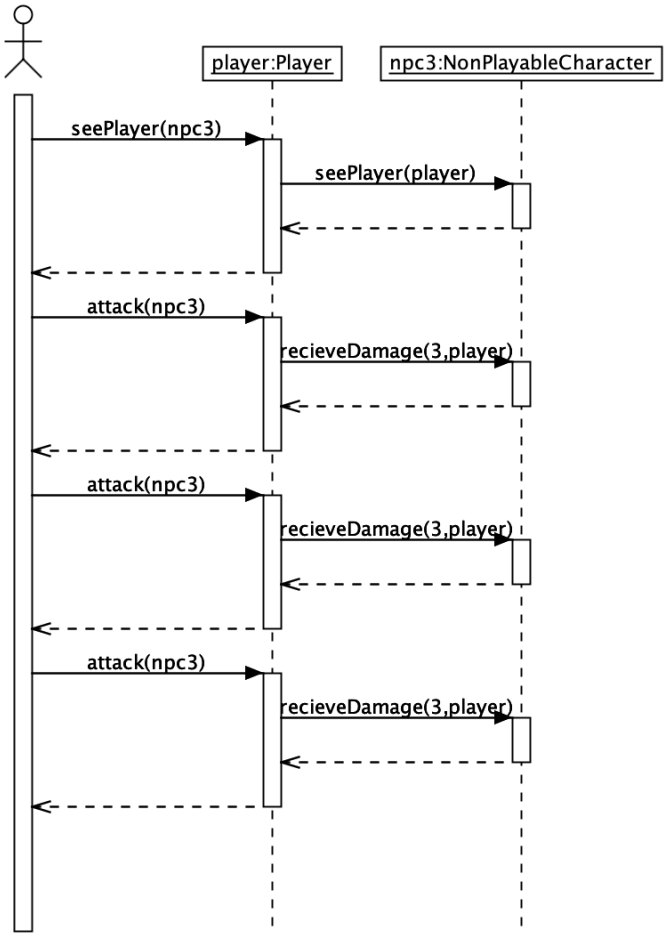

«enumeration»
PlayerState
neutral
panic
attack
Entity
health : Integer
name : String
level : Level
xPosition : Integer
yPosition : Integer
attack(target : Entity)
move(dx : Integer, dy : Integer)
seePlayer(entity : Entity)
loosePlayer(entity : Entity)
recieveDamage(damage : Integer, entity : Entity)
*
defender
*
attacker
«enumeration»
WeaponType
range
melee
magic
Player
NonPlayableCharacter
1
entity
1
weapon
Weapon
name : String
damage : Integer
type : WeaponType
«enumeration»
Level
recruit
basic
master
grandmaster
godtier

Initial State



Final State


sword:Weapon
name='Sword'
damage=3
type=#melee
weapon
entity
npc1:NonPlayableCharacter
health=30
name='Bone Crusher'
level=#basic
xPosition=10
yPosition=10
PlayerState=neutral
npc2:NonPlayableCharacter
health=15
name='Assasin from the East'
level=#master
xPosition=-10
yPosition=15
PlayerState=neutral
entity
weapon
fireball:Weapon
name='Fireball'
damage=7
type=#magic
dagger:Weapon
name='Dagger'
damage=1
type=#melee
weapon
entity
npc3:NonPlayableCharacter
health=-1
name='Common Criminal'
level=#recruit
xPosition=5
yPosition=5
PlayerState=death
bow:Weapon
name='Bow'
damage=3
type=#range
weapon
entity
player:Player
health=20
name='Brady'
level=#recruit
xPosition=0
yPosition=0
PlayerState=neutral


player:Player
npc3:NonPlayableCharacter
seePlayer(npc3)
seePlayer(player)
attack(npc3)
recieveDamage(3,player)
attack(npc3)
recieveDamage(3,player)
attack(npc3)
recieveDamage(3,player)

Commands.x

```
-- creating characters
!create player:Player
!set player.name := 'Brady'
!set player.xPosition := 0
!set player.yPosition := 0
!set player.health := 20
!set player.level := Level::recruit

!create npc1:NonPlayableCharacter
!set npc1.name := 'Bone Crusher'
!set npc1.xPosition := 10
!set npc1.yPosition := 10
!set npc1.health := 30
!set npc1.level := Level::basic

!create npc2:NonPlayableCharacter
!set npc2.name := 'Assasin from the East'
!set npc2.xPosition := -10
!set npc2.yPosition := 15
!set npc2.health := 15
!set npc2.level := Level::master

!create npc3:NonPlayableCharacter
!set npc3.name := 'Common Criminal'
!set npc3.xPosition := 5
!set npc3.yPosition := 5
!set npc3.health := 8
!set npc3.level := Level::recruit

!create bow:Weapon
!set bow.name := 'Bow'
!set bow.damage := 3
!set bow.type := WeaponType::range
!insert (player, bow) into entityWeapon

!create sword:Weapon
!set sword.name := 'Sword'
!set sword.damage := 3
!set sword.type := WeaponType::melee
!insert (npc1, sword) into entityWeapon

!create fireball:Weapon
```

```
!set fireball.name := 'Fireball'
!set fireball.damage := 7
!set fireball.type := WeaponType::magic
!insert (npc2, fireball) into entityWeapon

!create dagger:Weapon
!set dagger.name := 'Dagger'
!set dagger.damage := 1
!set dagger.type := WeaponType::melee
!insert (npc3, dagger) into entityWeapon

-- end create characters

!player.seePlayer(npc3)
!player.attack(npc3)
!player.attack(npc3)
!player.attack(npc3)
```

Model.use

```
/* Model created by Brady Cornett and Marcus Twichel
 * Advanced Software Engineering
 * Homework 3
 * 02/24/2020
 */
model ourFPS

 enum Level {recruit, basic, master, grandmaster, godtier}
 enum WeaponType {range, melee, magic}
 enum PlayerState {neutral, panic, attack}

 abstract class Entity
  attributes
   health: Integer
   name: String
   level: Level
   xPosition: Integer
   yPosition: Integer
  operations
   attack(target:Entity)
    begin
     target.recieveDamage(self.weapon.damage, self);
    end
   move(dx: Integer, dy: Integer)
    begin
     self.xPosition := self.xPosition + dx;
     self.yPosition := self.yPosition + dy;
    end
   seePlayer(entity:Entity)
    begin
     if not self.attacker->includes(entity)
     then
      insert (self, entity) into enemiesAssociation;
      entity.seePlayer(self)
     end
    end
   loosePlayer(entity:Entity)
    begin
     delete (self, entity) from enemiesAssociation;
    end
   recieveDamage(damage: Integer, entity:Entity)
    begin
     self.health := self.health - damage;
```

```
        if self.health < 0
        then
          delete (entity, self) from enemiesAssociation;
        end
      end
    statemachines
     psm PlayerState
      states
        startup:initial
        neutral
        attack
        panic
        death [health <= 0]
        -- find final state
      transitions
        startup -> neutral { create }
        -- if our weapon is better than opponent's, we attack
        neutral -> attack { seePlayer() [self.attacker.weapon.damage->sum() <
self.weapon.damage] }
        attack -> attack { recieveDamage() [health > 0] }
        panic -> panic { recieveDamage() [health > 0] }
        neutral -> panic { seePlayer() [self.attacker.weapon.damage->sum() >=
self.weapon.damage] }
        attack -> neutral { loosePlayer() [self.attacker->size() = 0] }
        panic -> neutral { loosePlayer() [self.attacker->size() = 0] }
        attack -> death { recieveDamage() [health <= 0] }
        panic -> death { recieveDamage() [health <= 0] }
        attack -> neutral { attack() [self.defender->size() = 0] }
        panic -> neutral { attack() [self.defender->size() = 0] }
        attack -> attack { attack() [self.defender->size() <> 0] }
        panic -> panic { attack() [self.defender->size() <> 0] }

      end

  end

  class Player < Entity
   attributes
   operations
  end


  class NonPlayableCharacter < Entity
```

```
    attributes
    operations
  end

  class Weapon
    attributes
      name: String
      damage: Integer
      type: WeaponType
  end

-- associations
association enemiesAssociation between
  Entity[0..*] role attacker
  Entity[0..*] role defender
end

association entityWeapon between
  Entity[1] role entity
  Weapon[1] role weapon
End
```