Calvin Liu

CS 32

Report

All virtual destructors must be there because the constructor takes in parameters

All doSomething() must be included because the super class Actor has the pure virtual function that will call all of them if I choose to call an actor’s doSomething

class StudentWorld;

class Actor : public GraphObject

{

public:

Actor(int imageID, int startX, int startY, StudentWorld\* object);

virtual void doSomething() = 0;

virtual ~Actor();

StudentWorld\* getWorld();

bool isDead();

void changetodead();

private:

StudentWorld\* key;

bool m\_status;

};

Actor(int imageID, int startX, int startY, StudentWorld\* object); needs to take in an imageID, startX, and startY to pass it to graphobject, and needs object to initialize key which will allow the actor to be a part of studentworld through a pointer.

virtual void doSomething() = 0; must be virtual so it calls all the derived classes doSomething which all do different things. Pure virtual because you can’t make an actor class and so you don’t have to implement actor class’ doSomething.

StudentWorld\* getWorld(); allows all my actors to have access to the studentworld and be a part of it.

isDead() shows if the things are dead

void changetodead() changes the status to dead

class Ship : public Actor

{

public:

Ship(StudentWorld\* object);

virtual ~Ship();

virtual void doSomething();

int currenthitpoints();

int damage(int k);

bool didItFire();

int shipEnergy();

void addTorpedoes(int k);

int numberOfTorpedoes();

void heal();

private:

int m\_hitpoints;

bool m\_didItFire;

int m\_numberOfTorpedoes;

};

Ship(StudentWorld\* object);

currenthitpoints() returns the health of the ship

damage() takes in a damage either, the amount from a collision, bullet, or torpedo

didItFire() shows if the projectile was from an alien or ship

shipEnergy() shows the health of the ship

addTorpedoes allows for the goodie to allow access to adding torpedoes

heal() is for the ship to heal once it gets an energyGoodie

class Star : public Actor

{

public:

Star(StudentWorld\* object);

virtual ~Star();

virtual void doSomething();

private:

};

star’s doSomething() only moves it down

class Alien : public Actor

{

public:

Alien(int imageID, int startX, int startY, StudentWorld\* object, int health);

virtual ~Alien();

virtual void doSomething() = 0;

void damage(int k);

int alienhealth();

private:

StudentWorld\* m\_ptrToWorld;

int m\_alienHealth;

};

All Aliens will have an id, an x, a y, a health, a pointer to the studentWorld

damage allows the alien to take in whatever damage is done to them

alienhealth() return the aliens health

class Nachling : public Alien

{

public:

Nachling(StudentWorld\* object, int HP);

virtual ~Nachling();

virtual void doSomething();

private:

bool everyOther;

};

Nachling has it’s own health it takes in

class WealthyNachling : public Alien

{

public:

WealthyNachling(StudentWorld\* object, int HP);

virtual ~WealthyNachling();

virtual void doSomething();

private:

int m\_state;

bool m\_malfunctioning;

bool m\_stillmalfunctioning;

int m\_malfunctCount;

};

WealthyNachling has it’s own health it takes in

The private members are used for the malfunctioning reason. They could be local variables. malfunctCount is for the at least 30 ticks wait.

class Smallbots : public Alien

{

public:

Smallbots(StudentWorld\* object, int HP);

virtual ~Smallbots();

virtual void doSomething();

bool justAttacked();

private:

bool m\_sEveryOther;

bool m\_justAttacked;

};

justAttacked() keeps track of whether or not it was just attacked. If it was just attacked then m\_justAttacked will be changed and flipped to keep track of if it just got attacked.

class Projectile : public Actor

{

public:

Projectile(int imageID, int startX, int startY, StudentWorld\* object, bool playerFired, int damagePoints);

virtual ~Projectile();

virtual void doSomething() = 0;

bool getPlayerFired();

bool didItFire();

int damagePoint();

private:

StudentWorld\* m\_ptrToWorld;

bool m\_playerFired;

int m\_damagePoints;

};

Projectile will have a bool and that indicates who shot it - The ship or the alien.

getPlayerFired() is too see if it is a projectile belonging to the ship or the alien.

didItFire() is for the every other instance kind of thing

damagePoint() return the amount of damage that was done depending on if it was a bullet or a torpedo

class Bullet : public Projectile

{

public:

Bullet(StudentWorld\* object, int startX, int startY, bool playerFired);

virtual ~Bullet();

virtual void doSomething();

private:

};

class Torpedo : public Projectile

{

public:

Torpedo(StudentWorld\* object, int startX, int startY, bool playerFired);

virtual ~Torpedo();

virtual void doSomething();

private:

};

the playerFired is there to show if the projectile was fired by the ship or the alien.

class Goodies : public Actor

{

public:

Goodies(int imageID, int startX, int startY, StudentWorld\* object);

~Goodies();

virtual void doSomething() = 0;

int everythird();

double goodieTickLifetime();

double ticksLeftToLive();

void resetthird();

void decrementticks();

private:

int m\_everythird;

double m\_goodieTickLifetime;

double m\_ticksLeftToLive;

};

everythird() to have the goodie move down after 3 ticks

goodieTickLifetime() returns the lifetime of the goodie

ticksLeftToLive() returns the tiks left until the goodie disappears

resetthird resets the everythird count to 0 so it can do it again at the end

decrementticks() decreases the ticks one by one

class FreeShipGoodie : public Goodies

{

public:

FreeShipGoodie(StudentWorld\* object, int startX, int startY);

virtual ~FreeShipGoodie();

virtual void doSomething();

private:

};

class EnergyGoodie : public Goodies

{

public:

EnergyGoodie(StudentWorld\* object, int startX, int startY);

virtual ~EnergyGoodie();

virtual void doSomething();

private:

};

class TorpedoGoodie : public Goodies

{

public:

TorpedoGoodie(StudentWorld\* object, int startX, int startY);

virtual ~TorpedoGoodie();

virtual void doSomething();

private:

};

Known Bugs:

I know my program crashes at random times

Sometimes the collision between the ship and an alien does not happen

The bullet and projectile does not execute the damage function.

Did not fix all of my projectiles on the screen count to be just aliens’

The alien does not come back to the screen when it goes off

Alien bullets are constructed in front of the alien, but destroyed right after or it goes it the opposite direction it is supposed to

Ambiguity:

I don’t think that we are suppose to include the player bullets into the count of the projectile onto the screen

The aliens sometimes went off screen to the right and left so I tried making it so the alien would move in the opposite direction in which the wall was