Milestone 4

shotgunrobot

2D platformer game written in Java.
https://github.com/rustbyte/shotgunrobot

Names	IDs
Felicia Santoro-Petti	6619657
Daniel Caterson	9746277
Amanda Juneau	6338518
Mark Karanfil	1526294
George Valergas	6095836

December 1, 2014
We have compiled and assessed this project a believe it to be of a reasonable size for a term project.

Summary of Project

Shotgun Robot is a 2D platformer game written in Java and hosted on Github. The primary contributor goes by the Github username *rustbyte*. He frequently codes live using a service called hitbox. The player assumes the role of a robot with a shotgun, whose purpose is to save humanity from extinction one person at a time, rescuing people as you fight off blood hungry zombies. The game takes place during a zombie apocalypse and does a decent job at ripping all of its concepts from Robocop, Terminator and similar franchises.

Refactorings

Change 1/3 – Move Powerup from extending Mob

Previously, the Powerup class extended from the Mob class just to inherit the move method in Mob. However, it also inherited the destructible interface even though the Powerup class is not a destructible entity, as well as the blockedX and blockedY attributes that Powerup never uses.

Powerup now extends Entity directly instead of Mob. There is now a move() method inside of the Powerup class that performs the same actions it did when using move from Mob, without the unneeded changes to the blockedX and blockedY attributes.

There now exists an interface called Moveable, which was created to make all entity objects implement this interface if that object needs to move in game. This is because not all objects that extent the Entity class use the move() method inherited from Entity, including Powerup. Therefore, the move() method was removed from the Entity class. All classes that implement move and extends Entity now implement the Moveable interface, including PowerUp.

Patch set 1/3

```
diff --git a/.gitignore b/.gitignore
index cb290e3..0f7ed73 100644
--- a/.gitignore
+++ b/.gitignore
@@ -1,2 +1,3 @@
 *.class
-*.jar
\ No newline at end of file
+*.jar
+/bin
diff --git a/src/com/rustbyte/BulletTrace.java b/src/com/rustbyte/BulletTrace.java
index d8b78d6..6cafe7d 100644
--- a/src/com/rustbyte/BulletTrace.java
+++ b/src/com/rustbyte/BulletTrace.java
@@ -68,10 +68,4 @@ public void render() {
                 }
         }

-        @Override
-        public void move() {
-                // TODO Auto-generated method stub
-
-        }
-
 }
diff --git a/src/com/rustbyte/Debris.java b/src/com/rustbyte/Debris.java
index 3086055..54bc297 100644
--- a/src/com/rustbyte/Debris.java
+++ b/src/com/rustbyte/Debris.java
@@ -2,7 +2,7 @@

 import java.util.Random;
```

```
-public class Debris extends Entity {
+public class Debris extends Entity implements Moveable {

        private int ticks;
        private int spriteX;
diff --git a/src/com/rustbyte/Entity.java b/src/com/rustbyte/Entity.java
index 2f48289..2a65e82 100644
--- a/src/com/rustbyte/Entity.java
+++ b/src/com/rustbyte/Entity.java
@@ -81,5 +81,4 @@ public void tick() {
                 }
        }
        public abstract void render() throws Exception;
-       public abstract void move();
 }
diff --git a/src/com/rustbyte/FloatingText.java b/src/com/rustbyte/FloatingText.java
index 02106b5..a8aed71 100644
--- a/src/com/rustbyte/FloatingText.java
+++ b/src/com/rustbyte/FloatingText.java
@@ -2,7 +2,7 @@

 import com.rustbyte.vector.Vector2;

-public class FloatingText extends Entity {
+public class FloatingText extends Entity implements Moveable {

        private String text;
        private Vector2 floatDir;
diff --git a/src/com/rustbyte/Grenade.java b/src/com/rustbyte/Grenade.java
index 18e3262..645b659 100644
--- a/src/com/rustbyte/Grenade.java
+++ b/src/com/rustbyte/Grenade.java
@@ -6,7 +6,7 @@
 import com.rustbyte.level.Tile;
 import com.rustbyte.vector.Vector2;

-public class Grenade extends Entity {
+public class Grenade extends Entity implements Moveable {
        private int damage = 100;
        private int fuseTimer = 200;
        private boolean detonated = false;
diff --git a/src/com/rustbyte/Mob.java b/src/com/rustbyte/Mob.java
index 6cfb0fb..a223fb7 100644
--- a/src/com/rustbyte/Mob.java
+++ b/src/com/rustbyte/Mob.java
@@ -6,7 +6,7 @@
 import com.rustbyte.level.Tile;
 import com.rustbyte.vector.Vector2;

-public abstract class Mob extends Entity implements Destructable {
+public abstract class Mob extends Entity implements Destructable, Moveable {
        public int hitpoints;
        protected int hurtTimer = 0;

diff --git a/src/com/rustbyte/Moveable.java b/src/com/rustbyte/Moveable.java
new file mode 100644
index 0000000..7086d9d
--- /dev/null
+++ b/src/com/rustbyte/Moveable.java
@@ -0,0 +1,8 @@
+package com.rustbyte;
+
+/**
+ * Interface which gives an entity the ability to move.
+ */
+public interface Moveable {
+       public void move();
+}
diff --git a/src/com/rustbyte/Particle.java b/src/com/rustbyte/Particle.java
index aa9b5e6..d3fd9a6 100644
```

```
--- a/src/com/rustbyte/Particle.java
+++ b/src/com/rustbyte/Particle.java
@@ -1,6 +1,6 @@
 package com.rustbyte;
 
-public class Particle extends Entity {
+public class Particle extends Entity implements Moveable {
        private int lifetime;
        private double px = 0.0;
        private double py = 0.0;
diff --git a/src/com/rustbyte/ParticleEmitter.java b/src/com/rustbyte/ParticleEmitter.java
index d7d91b9..b7f6cd2 100644
--- a/src/com/rustbyte/ParticleEmitter.java
+++ b/src/com/rustbyte/ParticleEmitter.java
@@ -58,10 +58,4 @@ public void tick() {
        public void render() {
 
        }
-
-       @Override
-       public void move() {
-               // TODO Auto-generated method stub
-
-       }
 }
diff --git a/src/com/rustbyte/Powerup.java b/src/com/rustbyte/Powerup.java
index 7c7d88c..4307968 100644
--- a/src/com/rustbyte/Powerup.java
+++ b/src/com/rustbyte/Powerup.java
@@ -2,7 +2,7 @@
 
 import com.rustbyte.Entity;
 
-public class Powerup extends Mob {
+public class Powerup extends Entity implements Moveable {
        public static final int POWERUP_TYPE_BATTERY = 1;
        public static final int POWERUP_TYPE_SHELLS = 2;
        public static final int POWERUP_TYPE_GRENADES = 3;
@@ -12,19 +12,21 @@
        private boolean pickedup = false;
        private int spawnTimer = 20;
        private int lifetime = 0;
+       private FlashEffect flashEffect;
+
        public Powerup(int x, int y, int w, int h, int type, Entity p, Game game) {
-               super(x,y,w,h, p, game );
+               super(x, y, w, h, p, game);
 
+               flashEffect = new FlashEffect(0xFFFFFF, 5, w, h);
                this.powerupType = type;
                int animID = 0;
 
                switch(this.powerupType) {
-               case POWERUP_TYPE_BATTERY: animID = animator.addAnimation(1, 0, 43, w,h, false, 1);
break;
-               case POWERUP_TYPE_SHELLS: animID = animator.addAnimation(1, 21, 43, w,h, false, 1);
break;
-               case POWERUP_TYPE_GRENADES: animID = animator.addAnimation(1, 21, 64, w,h, false,
1); break;
+                       case POWERUP_TYPE_BATTERY: animID = animator.addAnimation(1, 0, 43, w,h,
false, 1); break;
+                       case POWERUP_TYPE_SHELLS: animID = animator.addAnimation(1, 21, 43, w,h,
false, 1); break;
+                       case POWERUP_TYPE_GRENADES: animID = animator.addAnimation(1, 21, 64, w,h,
false, 1); break;
                }
 
-               this.hitpoints = 1;
                this.animator.setCurrentAnimation(animID);
        }
@@ -89,7 +91,24 @@ public void pickup() {
                         pickedup = true;
                 }
         }
+
+        /**
+         *       Slightly Modified version of Mob's move code. The Powerup class doesn't
+         *       rely on jumping, blockedX, or blockedY attributes to perform its
+         *       behaviour, so code using those variables was removed.
+         */
+        public void move() {
+                if(dirX < 0) facing = -1;
+                if(dirX > 0) facing = 1;
+
+                game.level.moveEntity(this, velocity.x, 0);
+                game.level.moveEntity(this, 0, velocity.y);
+
+                if(velocity.y != 0 )
+                        onground = false;
+
+                yy += velocity.y;
+                xx += velocity.x;
+        }

-        @Override
-        public void takeDamage(Entity source, int amount) { }
 }
```

Change 2/3 – Add encapsulation for Game.player to remove message chaining

There existed classes that extended Tile that required more information on the Player object to perform its functionality. To retrieve the Player, these Tile classes would have to retrieve the Game object from the Tile's level, and finally retrieve the Player object from Game. To remove this long message chain, a method getPlayer() was added to the Level class. The getPlayer() method localizes the chain of retrieves and returns the Player object needed to the caller.

The Player object inside of the Game class was changed from public to private, as well as its' getter and setter was added to the Game class. This ensures the encapsulation of a critical Game element that should never be exposed as public.

Removed some message chaining by making Game.player private. Level now uses game.getPlayer() method.

Within the Level class, Level has a getPlayer() method which delegates to the game object's getPlayer method.

```
public Player getPlayer() {
        return this.game.getPlayer();
}
```

In the future, the same concept could be used to encapsulate Level.game, which is currently accessed publicly.

Patch set 2/3

```
diff --git a/src/com/rustbyte/Game.java b/src/com/rustbyte/Game.java
index 3c051a6..f8b5d74 100644
--- a/src/com/rustbyte/Game.java
+++ b/src/com/rustbyte/Game.java
@@ -24,7 +24,7 @@

        public InputHandler input;
        public Level level;
-       public Player player;
+       private Player player;
        private List<Entity> entities = new ArrayList<Entity>();
        public int tickcount = 0;
        public int FPS = 0;
@@ -113,6 +113,7 @@ public Game(int wid, int hgt) {
                //addEntity(new BatBoss( ((level.width * 20) / 2) + 20, 100, 62,41, null,
this));
        }

+       public Player getPlayer() {return this.player;}

        private void renderPath() {
                Iterator<Entity> iter = entities.iterator();
diff --git a/src/com/rustbyte/Human.java b/src/com/rustbyte/Human.java
index e4fb8ab..a38e61a 100644
--- a/src/com/rustbyte/Human.java
+++ b/src/com/rustbyte/Human.java
@@ -43,7 +43,7 @@ public Human(int x, int y, int w, int h, Entity p, Game g) {

        private double distanceToPlayer() {
                Vector2 v1 = new Vector2(xx,yy);
-               Vector2 v2 = new Vector2(game.player.xx,
game.player.yy);
+               Vector2 v2 = new Vector2(game.getPlayer().xx, game.getPlayer().yy);
                return v1.sub(v2).length();
        }

diff --git a/src/com/rustbyte/Powerup.java b/src/com/rustbyte/Powerup.java
index 7c7d88c..e1e21b8 100644
--- a/src/com/rustbyte/Powerup.java
+++ b/src/com/rustbyte/Powerup.java
@@ -80,9 +80,9 @@ public void render() throws Exception {
        public void pickup() {
                if(!pickedup && spawnTimer == 0) {
                        switch(this.powerupType) {
-                       case POWERUP_TYPE_BATTERY: game.player.hitpoints += 20; break;
-                       case POWERUP_TYPE_SHELLS: game.player.shells += 10;  break;
-                       case POWERUP_TYPE_GRENADES: game.player.grenades += 5; break;
+                       case POWERUP_TYPE_BATTERY: game.getPlayer().hitpoints += 20; break;
+                       case POWERUP_TYPE_SHELLS: game.getPlayer().shells += 10;  break;
+                       case POWERUP_TYPE_GRENADES: game.getPlayer().grenades += 5; break;
                        }

                        pickupTimer = 200;
diff --git a/src/com/rustbyte/level/BossSpawnerTile.java
b/src/com/rustbyte/level/BossSpawnerTile.java
index 8dd0ca6..ee77242 100644
--- a/src/com/rustbyte/level/BossSpawnerTile.java
+++ b/src/com/rustbyte/level/BossSpawnerTile.java
@@ -71,7 +71,7 @@ public void init() {
        public void tick() {
                if( !activated ) {
                        // Check distance to player.
-                       Player p = level.game.player;
+                       Player p = level.getPlayer();
                        Vector2 playerPos = new Vector2(p.xx, p.yy);
                        Vector2 myPos = new Vector2((this.tx * 20) + 10, (this.ty * 20) + 10);
                        Vector2 delta = myPos.sub(playerPos);
diff --git a/src/com/rustbyte/level/Level.java b/src/com/rustbyte/level/Level.java
index b86a836..51b9817 100644
--- a/src/com/rustbyte/level/Level.java
+++ b/src/com/rustbyte/level/Level.java
@@ -10,6 +10,8 @@
 import com.rustbyte.Bitmap;
 import com.rustbyte.Entity;
 import com.rustbyte.Human;
+import com.rustbyte.Game;
+import com.rustbyte.Player;
```

```
 public class Level {
        public int width;
@@ -52,6 +54,9 @@ public Level(Bitmap bitmap, int tw, int th, Game g) {
                for(int i=0; i < (width * height);i++)
                        map[i].init();
        }
+       public Player getPlayer() {
+               return this.game.getPlayer();
+       }

        public void setPlayerSpawn(Tile t) {
                lastUnlockedSpawnLocation = t;
diff --git a/src/com/rustbyte/level/MobSpawnerTile.java
b/src/com/rustbyte/level/MobSpawnerTile.java
index eb92db0..639e0f2 100644
--- a/src/com/rustbyte/level/MobSpawnerTile.java
+++ b/src/com/rustbyte/level/MobSpawnerTile.java
@@ -89,7 +89,7 @@ public void init() {
        public void tick() {
                if( !activated ) {
                        // Check distance to player.
-                       Player p = level.game.player;
+                       Player p = level.getPlayer();
                        Vector2 playerPos = new Vector2(p.xx, p.yy);
                        Vector2 myPos = new Vector2((this.tx * 20) + 10, (this.ty * 20) + 10);
                        Vector2 delta = myPos.sub(playerPos);
```

Change 3/3 – Implement Abstract Factory Design Pattern for Mob

MobSpawnerTile objects are responsible for spawning the same Mob type after they've been initialized. However, the way this was being done was by using the mobName String attribute which indicated what Mob object to create. Instead of using a string variable, The creational logic is moved out of the Mob class and into Mob Factory classes that all extend MobFactory.

Each of these Factories has a spawnMob() method that returns a new object of the desired Mob subclass.

This makes it easy to add new MobSpawners to Tile objects. Just create a new Factory for each new Mob and add the factory to any Tile.

Patch set 3/3

```
diff --git a/src/com/rustbyte/BatBossMobFactory.java b/src/com/rustbyte/BatBossMobFactory.java
new file mode 100644
index 0000000..23d59a5
--- /dev/null
+++ b/src/com/rustbyte/BatBossMobFactory.java
@@ -0,0 +1,20 @@
+package com.rustbyte;
+
+public class BatBossMobFactory extends MobFactory {
+       private static BatBossMobFactory batBossMobFactory;
+
+       private BatBossMobFactory(){}
+
+       public static BatBossMobFactory getInstance(){
+               if(batBossMobFactory == null){
+                       batBossMobFactory = new BatBossMobFactory();
+               }
+               return batBossMobFactory;
+       }
+
+       @Override
```

```
+	public Mob spawnMob(int x, int y, int w, int h, Entity ent, Game game) {
+		return new BatBoss(x, y, 62, 41, ent, game);
+	}
+
+}
\ No newline at end of file
diff --git a/src/com/rustbyte/HumanMobFactory.java b/src/com/rustbyte/HumanMobFactory.java
new file mode 100644
index 0000000..5977a30
--- /dev/null
+++ b/src/com/rustbyte/HumanMobFactory.java
@@ -0,0 +1,21 @@
+package com.rustbyte;
+
+public class HumanMobFactory extends MobFactory {
+	private static HumanMobFactory humanMobFactory;
+
+	private HumanMobFactory(){}
+
+	public static HumanMobFactory getInstance(){
+		if(humanMobFactory == null){
+			humanMobFactory = new HumanMobFactory();
+		}
+		return humanMobFactory;
+	}
+
+	@Override
+	public Mob spawnMob(int x, int y, int w, int h, Entity ent, Game game) {
+		// TODO Auto-generated method stub
+		return new Human(x, y, w, h, ent, game);
+	}
+
+}
\ No newline at end of file
diff --git a/src/com/rustbyte/Mob.java b/src/com/rustbyte/Mob.java
index 6cfb0fb..b558a04 100644
--- a/src/com/rustbyte/Mob.java
+++ b/src/com/rustbyte/Mob.java
@@ -24,17 +24,7 @@ public Mob(double x, double y, int w, int h, Entity p, Game g) {
 		flashEffect = new FlashEffect(0xFFFFFF, 5, w,h);
 	}
 
-	public static Mob createMob(String mobType, int x, int y, int w, int h, Entity ent, Game
game) {
-
-		Mob result = null;
-		switch(mobType) {
-		case "HUMAN": result = new Human(x,y,w,h,ent,game); break;
-		case "ZOMBIE": result = new Zombie(x,y,w,h,ent,game); break;
-		case "SKELETON": result = new Skeleton(x,y,w,h,ent,game); break;
-		case "BATBOSS": result = new BatBoss(x,y,62,41,ent,game); break;
-		}
-		return result;
-	}
+
 
 	@Override
 	public void tick() {
diff --git a/src/com/rustbyte/MobFactory.java b/src/com/rustbyte/MobFactory.java
new file mode 100644
index 0000000..b1e783c
--- /dev/null
+++ b/src/com/rustbyte/MobFactory.java
@@ -0,0 +1,5 @@
+package com.rustbyte;
+
+public abstract class MobFactory {
+	public abstract Mob spawnMob(int x, int y, int w, int h, Entity ent, Game game);
+}
\ No newline at end of file
```

```
diff --git a/src/com/rustbyte/SkeletonMobFactory.java b/src/com/rustbyte/SkeletonMobFactory.java
new file mode 100644
index 0000000..0171827
--- /dev/null
+++ b/src/com/rustbyte/SkeletonMobFactory.java
@@ -0,0 +1,19 @@
+package com.rustbyte;
+
+public class SkeletonMobFactory extends MobFactory {
+        private static SkeletonMobFactory skeletonMobFactory;
+
+        private SkeletonMobFactory(){}
+
+        public static SkeletonMobFactory getInstance(){
+                if(skeletonMobFactory == null){
+                        skeletonMobFactory = new SkeletonMobFactory();
+                }
+                return skeletonMobFactory;
+        }
+
+        @Override
+        public Mob spawnMob(int x, int y, int w, int h, Entity ent, Game game) {
+                return new Skeleton(x, y, w, h, ent, game);
+        }
+}
\ No newline at end of file
diff --git a/src/com/rustbyte/ZombieMobFactory.java b/src/com/rustbyte/ZombieMobFactory.java
new file mode 100644
index 0000000..97c6e9b
--- /dev/null
+++ b/src/com/rustbyte/ZombieMobFactory.java
@@ -0,0 +1,21 @@
+package com.rustbyte;
+
+public class ZombieMobFactory extends MobFactory {
+        private static ZombieMobFactory zombieMobFactory;
+
+        private ZombieMobFactory(){}
+
+        public static ZombieMobFactory getInstance(){
+                if(zombieMobFactory == null){
+                        zombieMobFactory = new ZombieMobFactory();
+                }
+                return zombieMobFactory;
+        }
+
+        @Override
+        public Mob spawnMob(int x, int y, int w, int h, Entity ent, Game game) {
+                // TODO Auto-generated method stub
+                return new Zombie(x, y, w, h, ent, game);
+        }
+
+}
\ No newline at end of file
diff --git a/src/com/rustbyte/level/BossSpawnerTile.java
b/src/com/rustbyte/level/BossSpawnerTile.java
index 8dd0ca6..4ca1d0a 100644
--- a/src/com/rustbyte/level/BossSpawnerTile.java
+++ b/src/com/rustbyte/level/BossSpawnerTile.java
@@ -1,7 +1,9 @@
 package com.rustbyte.level;

 import java.util.Random;
+import com.rustbyte.BatBossMobFactory;
 import com.rustbyte.Mob;
+import com.rustbyte.MobFactory;
 import com.rustbyte.Player;
 import com.rustbyte.vector.Vector2;

@@ -14,7 +16,7 @@
```

```
        private int maxMobs = 1;
        private int numMobsSpawned = 0;
        private int spawnType = 0;
-       private String mobName;
+       private MobFactory mobFactory;

        public BossSpawnerTile(int type, int x, int y, int wid, int hgt, Level lev) {
                super(x, y, wid, hgt, lev);
@@ -24,14 +26,16 @@ public BossSpawnerTile(int type, int x, int y, int wid, int hgt, Level lev) {
                this.spawnType = type;

                switch(this.spawnType) {
-               case BOSS_SPAWN_TYPE_BAT: mobName = "BATBOSS"; break;
+               case BOSS_SPAWN_TYPE_BAT:
+                       mobFactory = BatBossMobFactory.getInstance();
+                       break;
                }
        }

        private void spawnMob() {

                if( nextMobTimer <= 0 && numMobsSpawned < maxMobs) {
-                       level.game.addEntity( Mob.createMob(mobName, (tx * 20) + 10, (ty * 20) +
10,20,20,null, level.game));
+                       level.game.addEntity(mobFactory.spawnMob((tx * 20) + 10, (ty * 20) +
10,20,20,null, level.game));
                        nextMobTimer = 60;
                        numMobsSpawned++;
                }
diff --git a/src/com/rustbyte/level/MobSpawnerTile.java
b/src/com/rustbyte/level/MobSpawnerTile.java
index eb92db0..bb66e81 100644
--- a/src/com/rustbyte/level/MobSpawnerTile.java
+++ b/src/com/rustbyte/level/MobSpawnerTile.java
@@ -1,8 +1,12 @@
 package com.rustbyte.level;

 import java.util.Random;
+import com.rustbyte.HumanMobFactory;
 import com.rustbyte.Mob;
+import com.rustbyte.MobFactory;
 import com.rustbyte.Player;
+import com.rustbyte.SkeletonMobFactory;
+import com.rustbyte.ZombieMobFactory;
 import com.rustbyte.vector.Vector2;

 public class MobSpawnerTile extends Tile {
@@ -26,16 +30,22 @@ public MobSpawnerTile(int type, int x, int y, int wid, int hgt, Level lev) {
                this.spawnType = type;

                switch(this.spawnType) {
-               case MOB_SPAWN_TYPE_HUMAN: mobName = "HUMAN"; break;
-               case MOB_SPAWN_TYPE_ZOMBIE: mobName = "ZOMBIE"; break;
-               case MOB_SPAWN_TYPE_SKELETON: mobName = "SKELETON"; break;
+               case MOB_SPAWN_TYPE_HUMAN:
+                       mobFactory = HumanMobFactory.getInstance();
+                       break;
+               case MOB_SPAWN_TYPE_ZOMBIE:
+                       mobFactory = ZombieMobFactory.getInstance();
+                       break;
+               case MOB_SPAWN_TYPE_SKELETON:
+                       mobFactory = SkeletonMobFactory.getInstance();
+                       break;
                }
        }

        private void spawnMob() {

                if( nextMobTimer <= 0 && numMobsSpawned < maxMobs) {
```

```
-                       level.game.addEntity( Mob.createMob(mobName, (tx * 20) + 10, (ty * 20) +
10,20,20,null, level.game));
+                       level.game.addEntity(mobFactory.spawnMob((tx * 20) + 10, (ty * 20) +
10,20,20,null, level.game));
                        nextMobTimer = 60;
                        numMobsSpawned++;
                }
@@ -50,7 +60,7 @@ public void init() {
                        rand.setSeed(level.game.tickcount);

                        for(int i = 0; i < 5 + rand.nextInt(5); i++) {
-                               level.game.addEntity(Mob.createMob(mobName, tx * 20, ty *
20,20,20,null, level.game));
+                               level.game.addEntity(mobFactory.spawnMob(tx * 20, ty *
20,20,20,null, level.game));
                        }
                }
```