

Mindustry



Mindustry is a sandbox tower defense game. Players can create elaborate supply chains of conveyor belts to feed ammo into their turrets, produce materials to use for building, and defend their structures from waves of enemies.

Github Repository : [Anuken/Mindustry\(github.com\)](https://github.com/Anuken/Mindustry)

DEMO Game : [Mindustry Classic by Anuke](#)

Wiki : https://mindustry.fandom.com/wiki/Mindustry_Wiki>About

Member

1.นาย ธนวัฒน์	สุขแก้ว	63015069
2.นาย ภูษิต	เสื่อโคร่ง	63015137
3.นาย วายุ	แสงพิทักษ์	63015161
4.นาย ศรายุทธ	พ่อค้า	63015165
5.นาย ศิริภักดิ์	โซจอหอ	63015171
6.นาย สุธี	สาระพันธ์	63015190

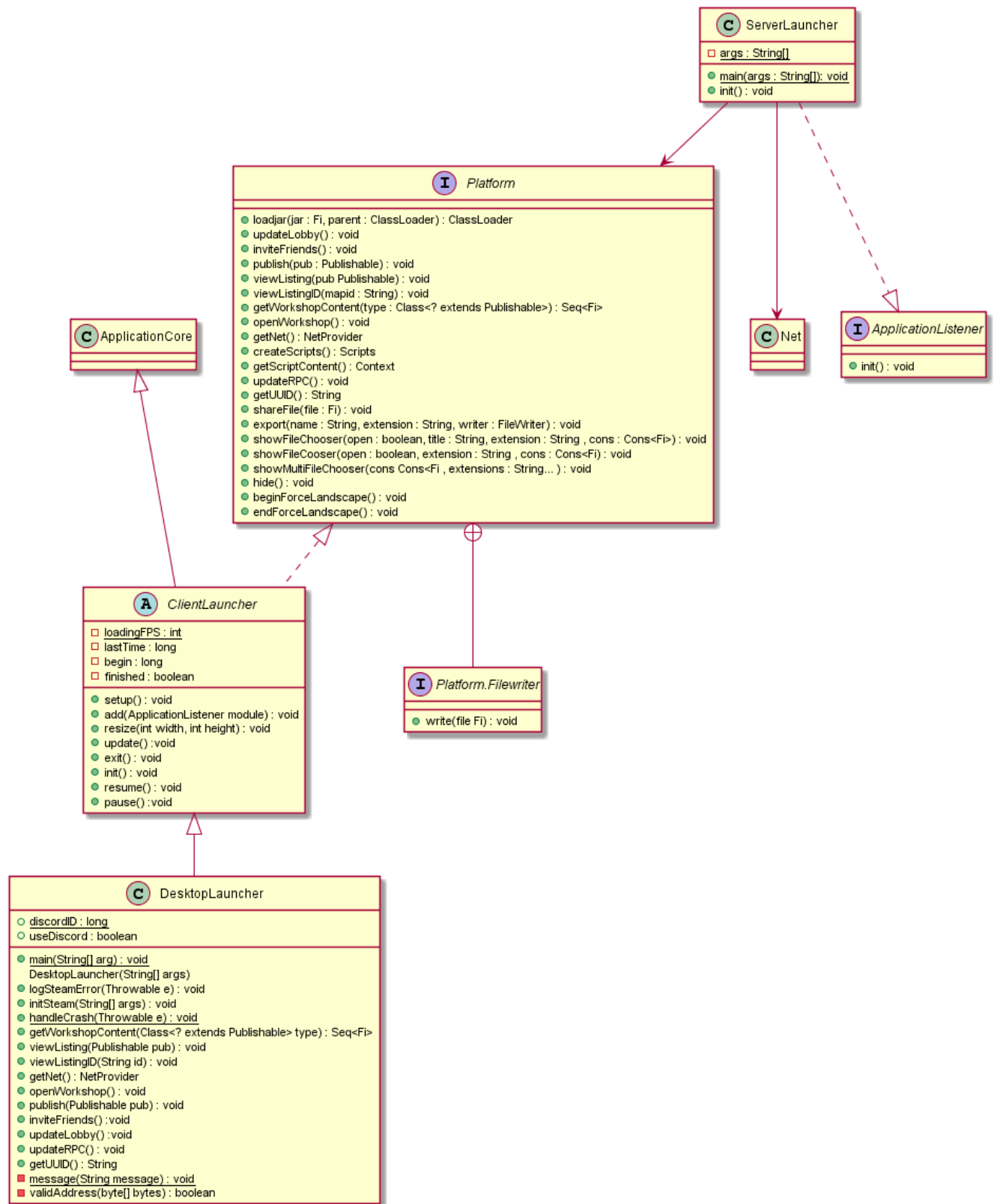
Architecture ของ Mindustry

1. Client-server

ตัวเกมมีการ host เพื่อเล่น online multiplayer / co-op ได้
โดยจะมีการ host 2 แบบคือ dedicated server และ local LAN
Ref [Servers - Mindustry](#)

โดย dedicated server architecture จะเป็น client-server
สามารถ support จำนวนผู้เล่นได้ดีกว่า local LAN
และยังใช้งานได้หลากหลายและมีประสิทธิภาพมากกว่าเนื่องจากมีคำสั่งมากมายเพื่อให้ผู้ดูแลระบบสามารถควบคุมได้มากขึ้น
และสามารถปรับเปลี่ยนให้เหมาะสมกับความต้องการของผู้ดูแลระบบได้อย่างง่ายดาย
Ref [Dedicated Server Architecture](#)





จุดอ่อนของ Architecture

จุดอ่อนของ Client-server Architecture คือ หาก server ล้มหยุดการทำงาน ระบบทั้งหมดจะหยุดการทำงานไปด้วย
จึงอาจทำให้ข้อมูลหรือ Progression ระหว่างเล่นสูญหายไป

วิธีแก้ปัญหาคือ ใช้งาน Auto save โดยกำหนด interval เวลาในการ Auto save เวลาเซฟล้มก็จะทำให้ข้อมูลสูญหายน้อยที่สุด(สูญหายไปบางช่วงส่วนมากเรียกว่า rollback) จนถึง ไม่เสียหาย

ref.[How to Configure Autosaving of Your Minecraft World - Knowledgebase - Shockbyte](#)

Quality Attributes ของ Mindustry

1. Portability : ตัวเกมมีการ support ถึง 5 platform ได้แก่ Windows, MacOS, Linux, Andriod, iOS

[Getting Started - Mindustry Documentation \(mindustrymodders.gitlab.io\)](https://mindustrymodders.gitlab.io)

Getting Started

Getting started with Mindustry is easy. This article covers how to install Mindustry on different platforms and situations.

Typical Setup

This is your typical, run-of-the-mill setup process.

Desktop

1. Visit the game's itch.io page, then download a copy of the game from there.
2. Once the `.zip` file is downloaded, unzip it. Usually that is just done by opening it like normal. When it's done, navigate to the folder it extracted into.
3.
 1. **Windows:** simply open `desktop-release.exe`.
 2. **MacOS:** run `Mindustry.app`.
 3. **Linux:** run `Mindustry`.

If you have JRE already installed (which is recommended), you can also run `desktop-release.jar` on Windows and Linux.

Android

This particular section covers how to install Mindustry's **latest builds** on Android. You might be here because you don't know what sort of Mindustry everyone's talking about on the Discord, or heard of `build xx`. If so, please read on.

1. Open Google Play Store and visit Mindustry's page.
2. Scroll down a bit and notice the "Become a beta tester" section. Press "Join now" and confirm. Fully signing up will take a while, so be patient.
3. The green button will then say "Update". Update and enjoy!

iOS

The latest released builds are available on Apple TestFlight.

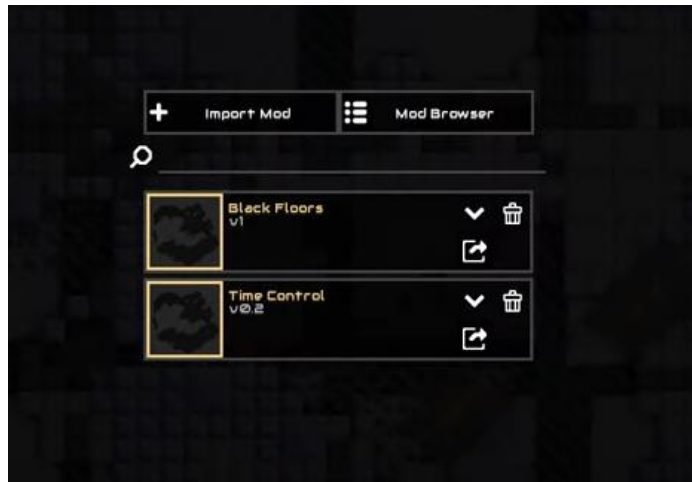
2. Interoperability : ตัวเกมสามารถ cross-platform ได้ โดยอิงจากข้อแรก
ที่เกมสามารถเล่นได้หลาย platform ทำให้ตัวเกมสามารถเล่นด้วยกันข้าม platform
ได้

[Frequently Asked Questions - Mindustry Wiki \(mindustrygame.github.io\)](https://github.com/mindustrygame/mindustrygame/wiki/Frequently-Asked-Questions)

3. Modifiability : มี class เช็ค event ต่างๆเพื่อให้สามารถนำไปสร้าง mod ใหม่ได้

[Scripting - Mindustry Wiki \(mindustrygame.github.io\)](https://github.com/mindustrygame/mindustrygame/wiki/Scripting)

4. Usability : ตัวเกมมี ui support ให้ลง mod ได้ง่ายโดยเราสามารถ Browse หา mod
ที่ต้องการได้



หน้าต่างของตัว Browser ที่ใช้หา mod และ install



Design Pattern ของ Mindustry

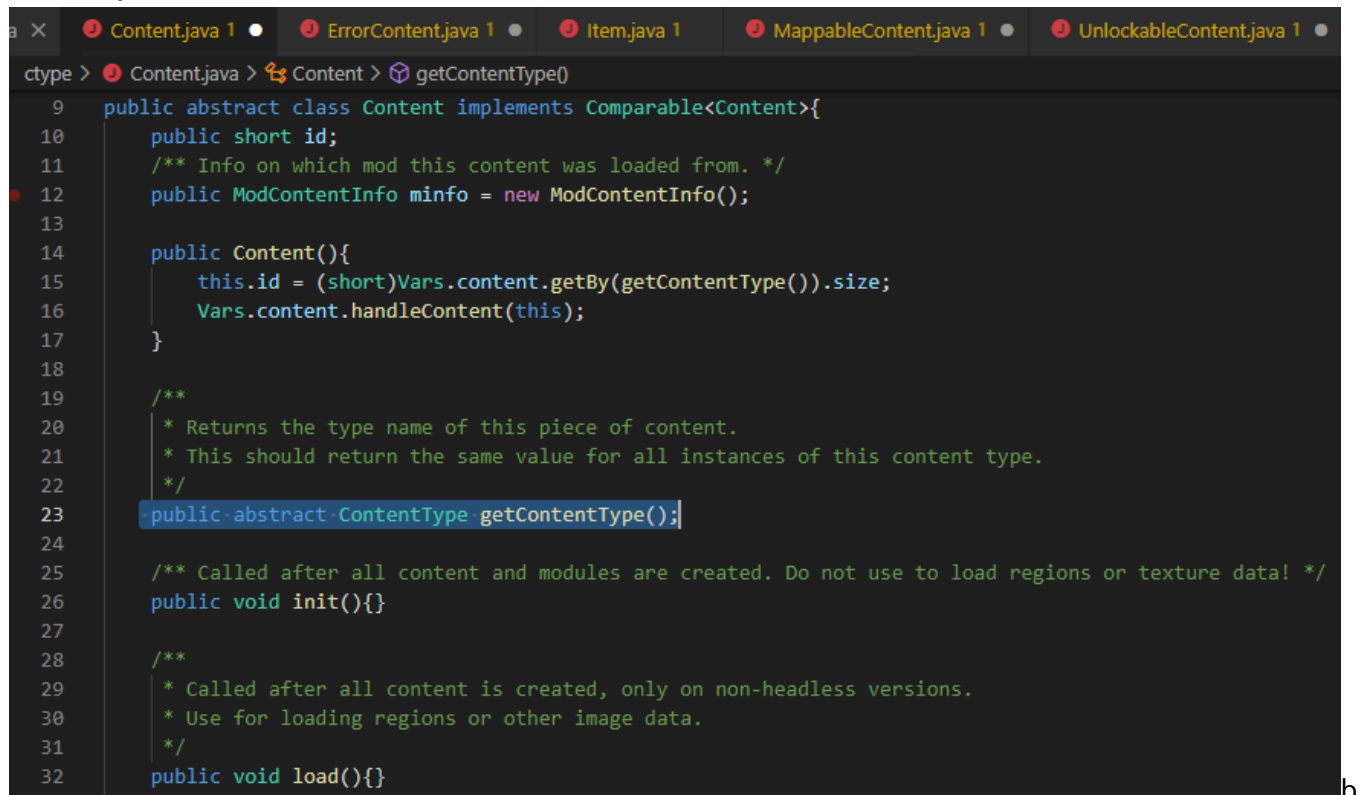
1.Factory Method

ในเกม Mindustry นั้นจะมีการสร้าง Object content ต่างๆของเกมขึ้นมามากมายและ content แต่ละตัวก็จะมี Type ที่แตกต่างกันไป ซึ่งในส่วนนี้ ได้มีการใช้ Factory Method เข้ามาช่วยในการระบุ Type ของ content ที่ทำการสร้างขึ้น

จาก code ตัวอย่าง ในไฟล์ Content.java มี abstract method สร้างไว้ชื่อ getContent() เพื่อดูค่า ContentType เพื่อให้ subclass ที่นำไปใช้ระบุค่า ContentType ของตัว Object ที่ subclass นั้นสร้างขึ้น

Source Code

Content.java [link](#)



```
Content.java > Content > getContent()
9 public abstract class Content implements Comparable<Content>{
10     public short id;
11     /** Info on which mod this content was loaded from. */
12     public ModContentInfo minfo = new ModContentInfo();
13
14     public Content(){
15         this.id = (short)Vars.content.getBy(getContentType()).size;
16         Vars.content.handleContent(this);
17     }
18
19     /**
20      * Returns the type name of this piece of content.
21      * This should return the same value for all instances of this content type.
22      */
23     public abstract ContentType getContentType();
24
25     /** Called after all content and modules are created. Do not use to load regions or texture data! */
26     public void init(){}
27
28     /**
29      * Called after all content is created, only on non-headless versions.
30      * Use for loading regions or other image data.
31      */
32     public void load(){}
b
```

ErrorContent.java [link](#)

```
Formation.java  Content.java 1  ErrorContent.java 1  MappableContent.java 1  UnlockableContent.java 1

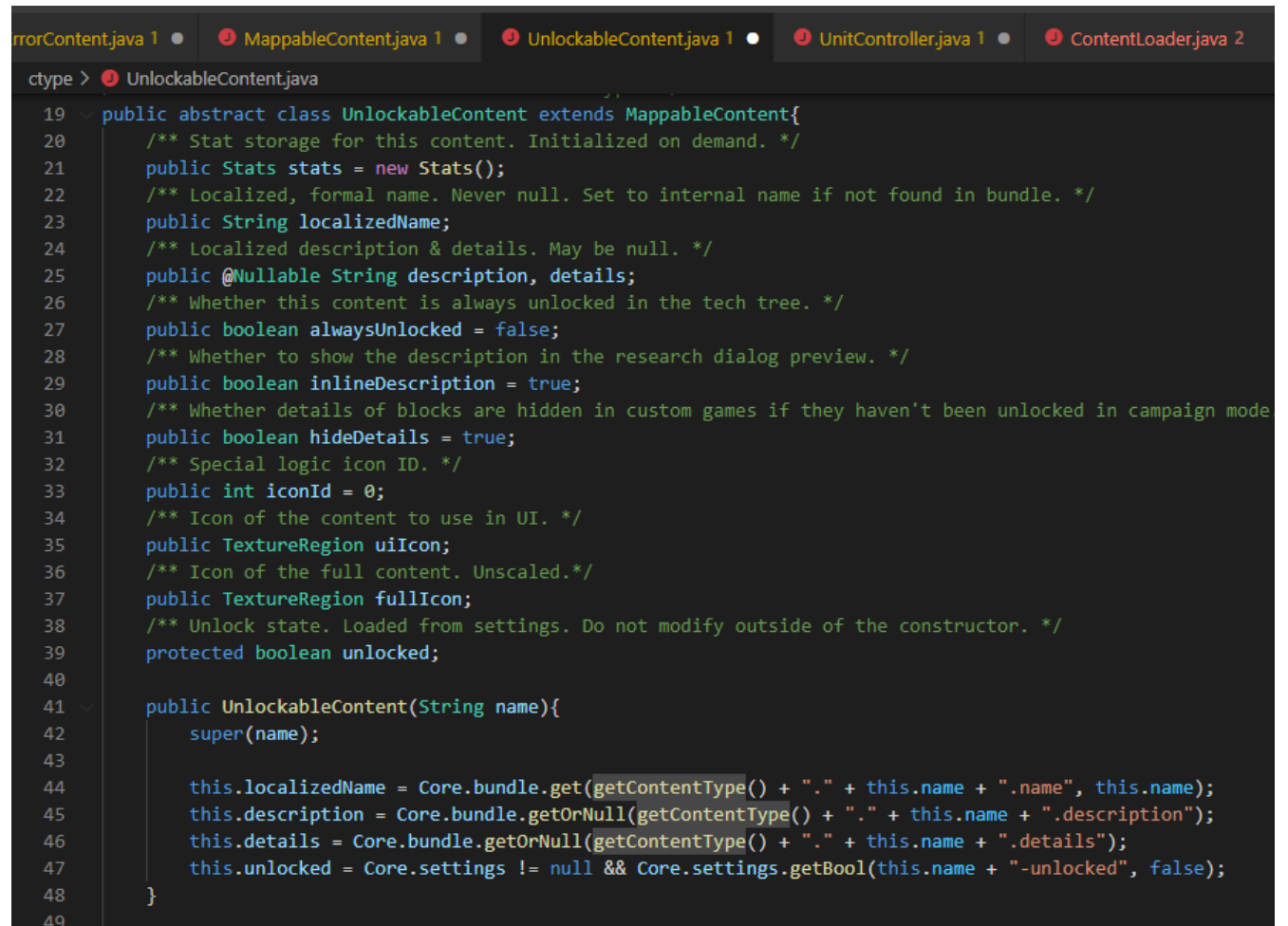
type > ErrorContent.java > ErrorContent
1  package mindustry.type;
2
3  import mindustry.ctype.*;
4
5  /** Represents a blank type of content that has an error. Replaces anything that failed to parse. */
6  public class ErrorContent extends Content{
7      @Override
8      public ContentType getContentType(){
9          return ContentType.error;
10     }
11 }
```

MappableContent.java [link](#)

```
ErrorContent.java 1  MappableContent.java 1  UnlockableContent.java 1  UnitController.j

ctype > MappableContent.java > MappableContent
1  package mindustry.ctype;
2
3  import mindustry.*;
4
5  public abstract class MappableContent extends Content{
6      public final String name;
7
8      public MappableContent(String name){
9          this.name = Vars.content.transformName(name);
10         Vars.content.handleMappableContent(this);
11     }
12
13     @Override
14     public String toString(){
15         return name;
16     }
17 }
```


UnlockableContent.java [link](#)



```
19 public abstract class UnlockableContent extends MappableContent{
20     /** Stat storage for this content. Initialized on demand. */
21     public Stats stats = new Stats();
22     /** Localized, formal name. Never null. Set to internal name if not found in bundle. */
23     public String localizedName;
24     /** Localized description & details. May be null. */
25     public @Nullable String description, details;
26     /** Whether this content is always unlocked in the tech tree. */
27     public boolean alwaysUnlocked = false;
28     /** Whether to show the description in the research dialog preview. */
29     public boolean inlineDescription = true;
30     /** Whether details of blocks are hidden in custom games if they haven't been unlocked in campaign mode */
31     public boolean hideDetails = true;
32     /** Special logic icon ID. */
33     public int iconId = 0;
34     /** Icon of the content to use in UI. */
35     public TextureRegion uiIcon;
36     /** Icon of the full content. Unscaled.*/
37     public TextureRegion fullIcon;
38     /** Unlock state. Loaded from settings. Do not modify outside of the constructor. */
39     protected boolean unlocked;
40
41     public UnlockableContent(String name){
42         super(name);
43
44         this.localizedName = Core.bundle.get(getContentType() + "." + this.name + ".name", this.name);
45         this.description = Core.bundle.getOrNull(getContentType() + "." + this.name + ".description");
46         this.details = Core.bundle.getOrNull(getContentType() + "." + this.name + ".details");
47         this.unlocked = Core.settings != null && Core.settings.getBool(this.name + "-unlocked", false);
48     }
49 }
```

BulletType.java [link](#)

```

BulletType.java 6 X  ErrorContent.java 1  MappableContent.java 1  UnlockableContent.java 1
entities > bullet > BulletType.java > BulletType > getContentType()

25  public class BulletType extends Content implements Cloneable{
26      /** Lifetime in ticks. */
27      public float lifetime = 40f;
28      /** Speed in units/tick. */
29      public float speed = 1f;
30      /** Direct damage dealt on hit. */
31      public float damage = 1f;
32      /** Hitbox size. */
33      public float hitSize = 4;
34      /** Clipping hitbox. */
35      public float drawSize = 40f;
36      /** Drag as fraction of velocity. */
37      public float drag = 0f;
38      /** Whether to pierce units. */

```

```

BulletType.java 6 X  ErrorContent.java 1  MappableContent.java 1  UnlockableContent.java 1  UnitContent.java 1
entities > bullet > BulletType.java > BulletType > getContentType()

433
434  @Override
435  public ContentType getContentType(){
436      return ContentType.bullet;
437  }
438
439  public Bullet create(Teamc owner, float x, float y, float angle){
440      return create(owner, owner.team(), x, y, angle);
441  }
442
443  public Bullet create(Entityc owner, Team team, float x, float y, float angle){
444      return create(owner, team, x, y, angle, velocitySc1: 1f);
445  }
446

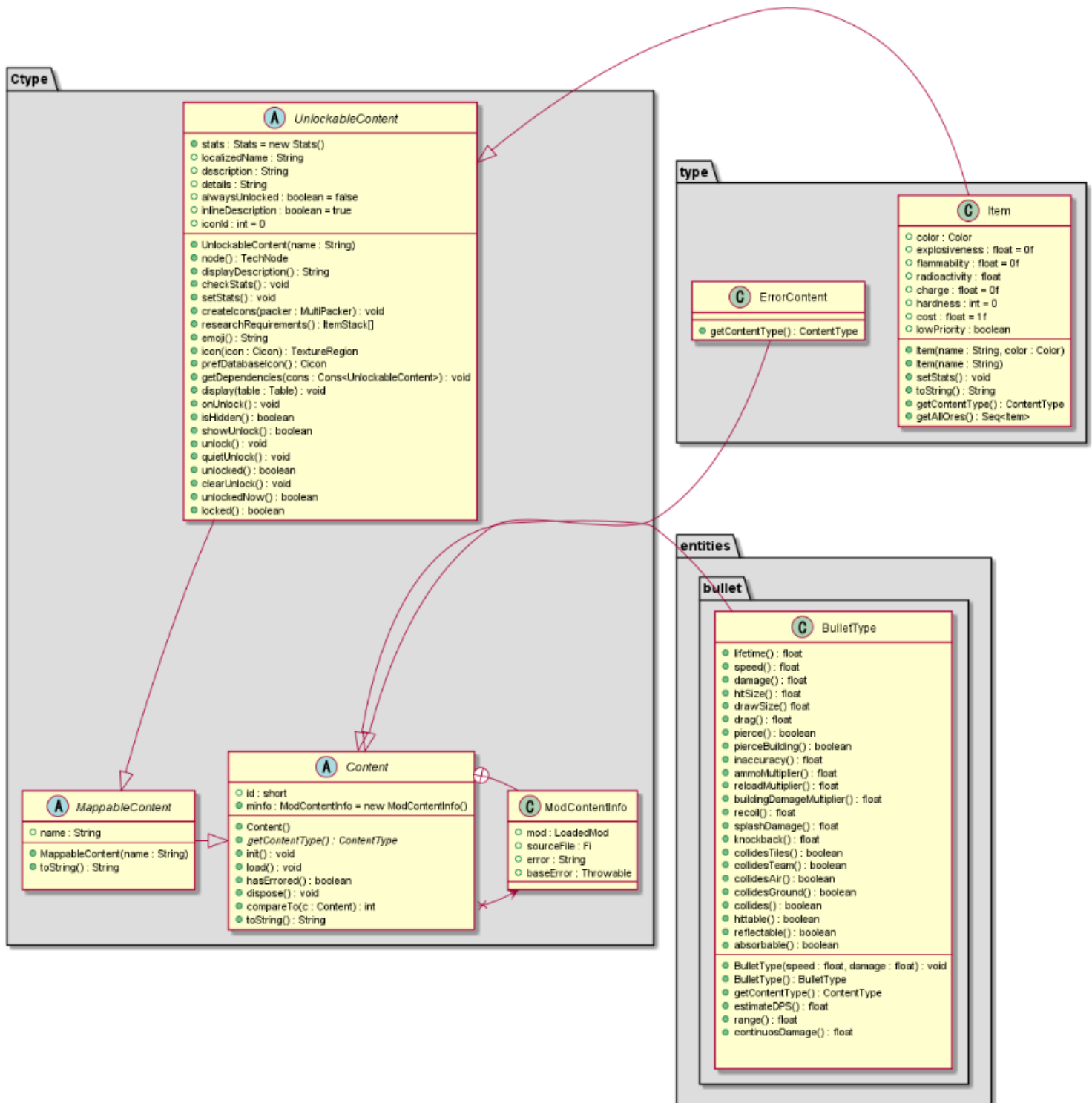
```

Item.java [link](#)

```
Item.java 1 x MappableContent.java 1 UnlockableContent.java 1 UnitController.java 1
type > Item.java > Item
9 import static mindustry.Vars.*;
10
11 public class Item extends UnlockableContent{
12     public Color color;
13
14     /** how explosive this item is. */
15     public float explosiveness = 0f;
16     /** flammability above 0.3 makes this eligible for item burners. */
17     public float flammability = 0f;
18     /** how radioactive this item is. */
19     public float radioactivity;
20     /** how electrically potent this item is. */
21     public float charge = 0f;
22     /** drill hardness of the item */
23     public int hardness = 0;
24 }

Item.java 1 x MappableContent.java 1 UnlockableContent.java 1 UnitController.java 1 Conte
type > Item.java > Item > getContentType()
54 @Override
55 public ContentType getContentType(){
56     return ContentType.item;
57 }
58
59 /** Allocates a new array containing all items that generate ores. */
60 public static Seq<Item> getAllOres(){
61     return content.blocks().select(b -> b instanceof OreBlock).map(b -> b.itemDrop);
62 }
63 }
64
```

UML Class Diagram



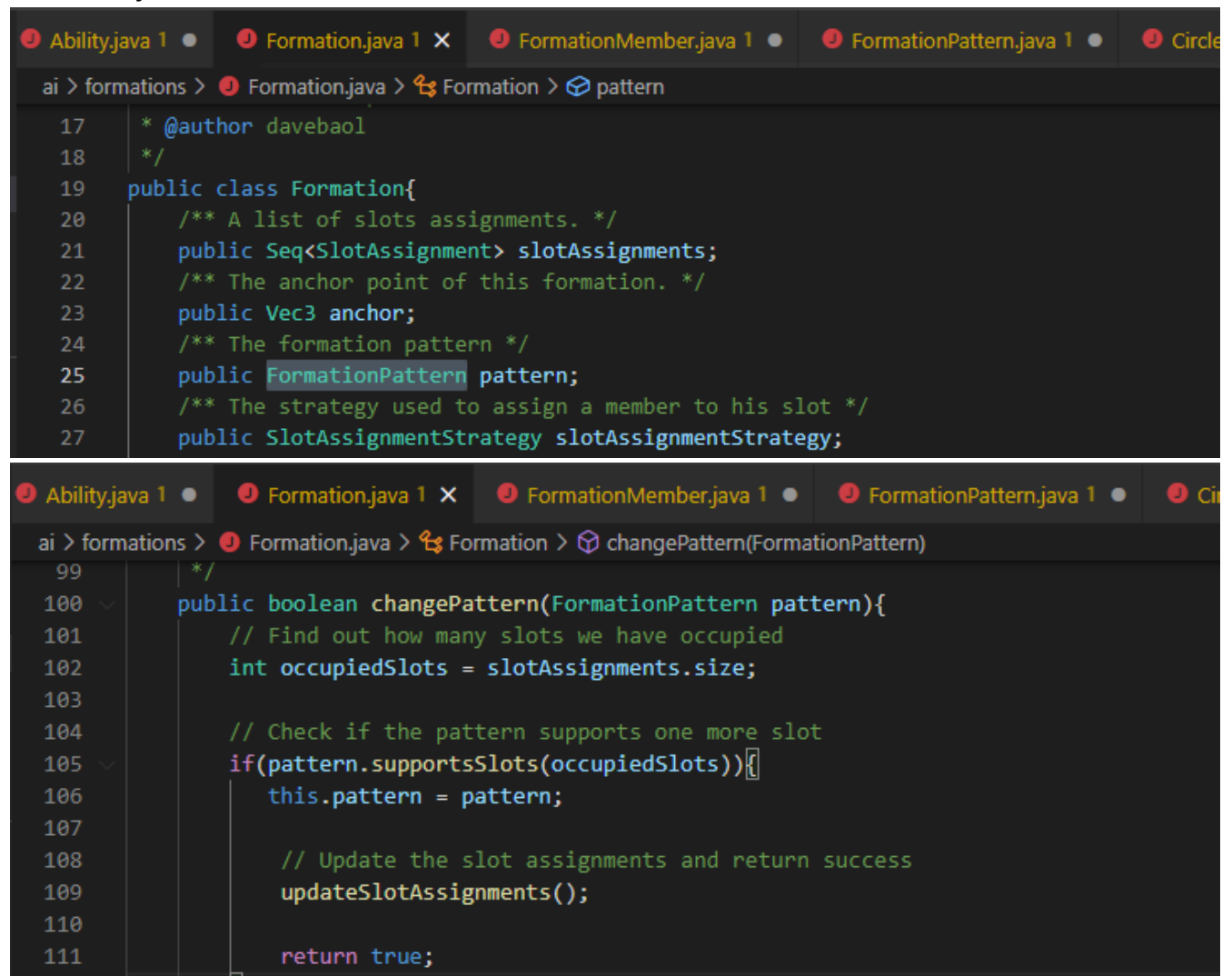
2. Strategy

ในเกม Mindustry มีการแยกการคำนวณกันในบางส่วน จึงมีการนำ Strategy มาใช้งานโดยสร้างในส่วนของ Algorithm ในการคำนวณแยก class กัน และให้ตัว client เป็นคนกำหนดว่าจะสร้าง object เพื่อเอาไว้ทำงานจาก class ไหน และทำให้สามารถเปลี่ยนการทำงานได้ตลอดเวลา ขณะ runtime

ในไฟล์ Formation.java นั้นมีการเรียกใช้งาน FormationPattern.java ในส่วนของ method calculateSlotLocation ซึ่งจะมีการแยก Class สำหรับการคำนวณของ method นี้ออกจากัน จากโค้ดตัวอย่างมีการแบ่งเป็น 2 class ได้แก่ CircleFormation.java และ SquareFormation.java

Source Code

Formation.java [link](#)



```
17  * @author davebaol
18  */
19  public class Formation{
20      /** A list of slots assignments. */
21      public Seq<SlotAssignment> slotAssignments;
22      /** The anchor point of this formation. */
23      public Vec3 anchor;
24      /** The formation pattern */
25      public FormationPattern pattern;
26      /** The strategy used to assign a member to his slot */
27      public SlotAssignmentStrategy slotAssignmentStrategy;

99  */
100 public boolean changePattern(FormationPattern pattern){
101     // Find out how many slots we have occupied
102     int occupiedSlots = slotAssignments.size;
103
104     // Check if the pattern supports one more slot
105     if(pattern.supportsSlots(occupiedSlots)){
106         this.pattern = pattern;
107
108         // Update the slot assignments and return success
109         updateSlotAssignments();
110
111         return true;
112     }
```

```
Formation.java 1 x UnitController.java 1 PlayerComp.java 1 AIController.java 3 For
ai > formations > Formation.java > Formation > updateSlots()
192 SlotAssignment slotAssignment = slotAssignments.get(i);
193
194 // Retrieve the location reference of the formation member to calculate
195 Vec3 relativeLoc = slotAssignment.member.formationPos();
196 float z = relativeLoc.z;
197
198 // Ask for the location of the slot relative to the anchor point
199 pattern.calculateSlotLocation(relativeLoc, slotAssignment.slotNumber);
200
201 // Transform it by the anchor point's position and orientation
```

FormationPattern.java [link](#)

```
Ability.java 1 FormationMember.java 1 FormationPattern.java 1 CircleFormation.java 1
ai > formations > FormationPattern.java > FormationPattern > spacing
11 @author DaveDoo1
12 */
13 public abstract class FormationPattern{
14     public int slots;
15     /** Spacing between members. */
16     public float spacing = 20f;
17
18     /** Returns the location of the given slot index. */
19     public abstract Vec3 calculateSlotLocation(Vec3 out, int slot);
20
21     /**
22      * Returns true if the pattern can support the given number of slots
23      * @param slotCount the number of slots
24      * @return {@code true} if this pattern can support the given number of slot
25      */
26     public boolean supportsSlots(int slotCount){
27         return true;
28     }
29 }
30
```

CircleFormation.java [link](#)

```
Ability.java 1 • FormationMember.java 1 • FormationPattern.java 1 • CircleFormation.java 1 • SquareForm
ai > formations > patterns > CircleFormation.java > ...
6
7 public class CircleFormation extends FormationPattern{
8
9     @Override
10    public Vec3 calculateSlotLocation(Vec3 outLocation, int slotNumber){
11        if(slots > 1){
12            float angle = (360f * slotNumber) / slots + (slots == 8 ? 22.5f : 0);
13            float radius = spacing / (float)Math.sin(180f / slots * Mathf.degRad);
14            outLocation.set(Angles.trnsx(angle, radius), Angles.trnsy(angle, radius), angle);
15        }else{
16            outLocation.set(0, spacing * 1.1f, 360f * slotNumber);
17        }
18
19        return outLocation;
20    }
21
22 }
23
```

SquareFormation.java [link](#)

```
Ability.java 1 • SquareFormation.java 1 • Formation.java 1 • FormationMember.java 1 • FormationPat
ai > formations > patterns > SquareFormation.java > SquareFormation > calculateSlotLocation(Vec3, int)
6
7 public class SquareFormation extends FormationPattern{
8
9     @Override
10    public Vec3 calculateSlotLocation(Vec3 out, int slot){
11        //side of each square of formation
12        int side = Mathf.ceil(Mathf.sqrt(slots + 1));
13        int cx = slot % side, cy = slot / side;
14
15        //don't hog the middle spot
16        if(cx == side / 2 && cy == side / 2 && (side % 2) == 1){
17            slot = slots;
18
19            cx = slot % side;
20            cy = slot / side;
21        }
22
23        return out.set(cx - (side / 2f - 0.5f), cy - (side / 2f - 0.5f), 0).scl(spacing);
24    }
25
26 }
```

UML Class Diagram

formations

Formation

- slotAssignments : Seq<SlotAssignment>
- anchor : Vec3
- slotAssignmentsStrategy : SlotAssignmentStrategy
- motionModerator : FormationModerator
- positionOffset : Vec3
- orientationMatrix : Mat
- driftOffset : Vec3
- pattern : FormationPattern

- Formation(anchor : Vec3, pattern : FormationPattern)
- updateSlotAssignmnet() : void
- changePattern(pattern : FormationPattern) : boolean
- addMembers(members : iterable<? extends FormationMember>) : int
- addMembers(member : FormationMember) : int
- getSlotAssignmnetAt(index : int) : SlotAssignment
- getSlotAssignmnetCount() : int
- updateSlots() : void

patterns

CircleFormation

- angleOffset : float
- calculateSlotLocation(out : Vec3, slot : int) : Vec3

SquareFormation

- calculateSlotLocation(out : Vec3, slot : int) : Vec3

FormationPattern

- slot : int
- spacing : Float
- calculateSlotLocation(out : Vec3, slot : int) : Vec3
- supportsSlots(slotCount : int) : boolean



3. Flyweight

ในเกม Mindustry มีสร้าง Object ที่ต้องมีการสร้างเยอะๆ เช่น Bullet จึงมีการนำตัว Flyweight มาช่วยในการสร้างโดยมีการแยกเก็บค่าต่างๆที่เหมือนกันแยกไว้อีกคลาสแทนที่จะเก็บไว้ใน Object แต่ละตัว เพื่อช่วยในการลดขนาดการใช้งาน Memory

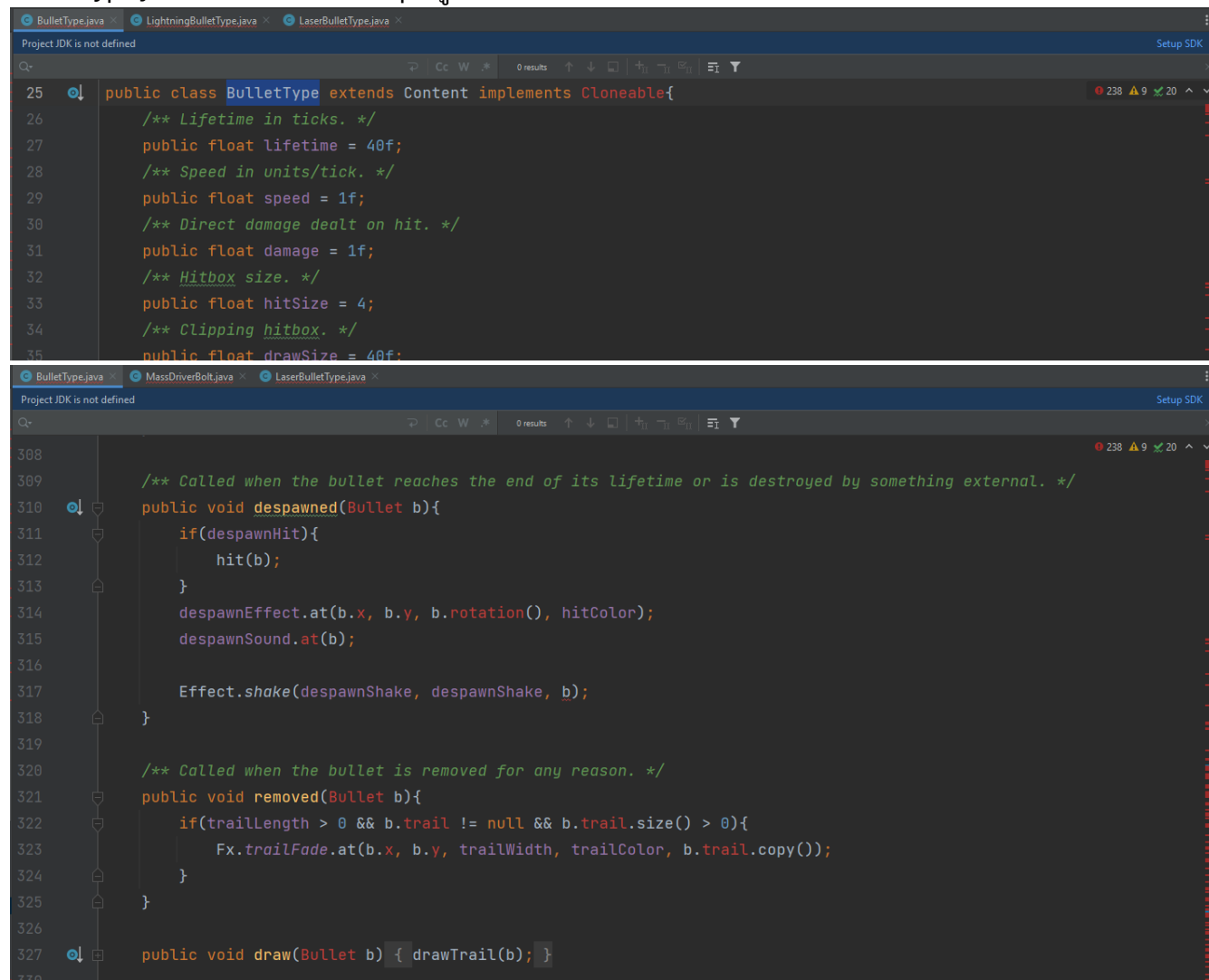
จากโค้ดมี BulletType.java เป็นตัวเก็บ extrinsic state และมี Weapon.java function shoot (322 - 372 เป็นตัวสร้าง object)

function bullet (375 - 382 เก็บ intrinsic state)

ตอนนี้มองว่า class Bullet นั้นไม่เป็น intrinsic state ไม่มีการเก็บ position ของกระสุนมีเพียงการสร้าง object ต่างๆอย่างเดียว

Source Code

BulletType.java [link](#) function ต่างๆจะถูกประกาศบรรทัดที่ 180 เป็นต้นไป



```
25 public class BulletType extends Content implements Cloneable{
26     /** Lifetime in ticks. */
27     public float lifetime = 40f;
28     /** Speed in units/tick. */
29     public float speed = 1f;
30     /** Direct damage dealt on hit. */
31     public float damage = 1f;
32     /** Hitbox size. */
33     public float hitSize = 4;
34     /** Clipping hitbox. */
35     public float drawSize = 40f;
36
37     /** Called when the bullet reaches the end of its lifetime or is destroyed by something external. */
38     public void despawned(Bullet b){
39         if(despawnHit){
40             hit(b);
41         }
42         despawnEffect.at(b.x, b.y, b.rotation(), hitColor);
43         despawnSound.at(b);
44
45         Effect.shake(despawnShake, despawnShake, b);
46     }
47
48     /** Called when the bullet is removed for any reason. */
49     public void removed(Bullet b){
50         if(trailLength > 0 && b.trail != null && b.trail.size() > 0){
51             Fx.trailFade.at(b.x, b.y, trailWidth, trailColor, b.trail.copy());
52         }
53     }
54
55     public void draw(Bullet b) { drawTrail(b); }
```

Weapon.java 375 - 382 เก็บ intrinsic state [link](#)

```
Fires.java × Bullets.java × BulletType.java × FireBulletType.java × LaserBulletType.java × PointDefenseWeapon.java × RepairBeamWeapon.java × Weapon.java × WeaponMount.java ×
Project JDK is not defined Setup SDK
374
375     protected Bullet bullet(Unit unit, float shootX, float shootY, float angle, float lifesc1){
376         float xr = Mathf.range(xRand);
377
378         return bullet.create(unit, unit.team,
379             x: shootX + Angles.trnsx(angle, 0, xr),
380             y: shootY + Angles.trnsy(angle, 0, xr),
381             angle, velocityScd: (1f - velocityRnd) + Mathf.random(velocityRnd), lifesc1);
382     }
383
```

.Bullet.java [link](#)

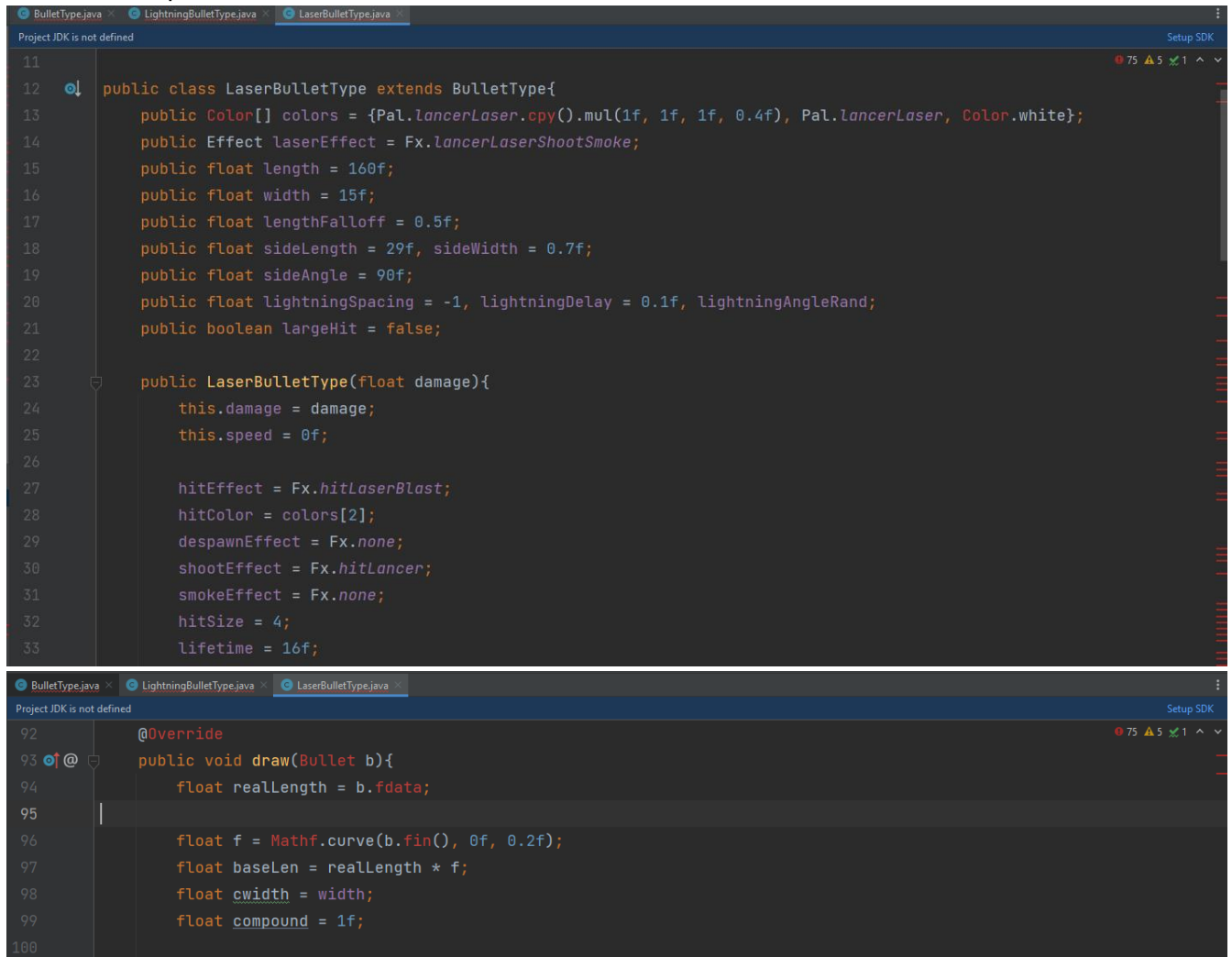
```
Fires.java × Bullets.java × BulletType.java × FireBulletType.java × LaserBulletType.java × PointDefenseWeapon.java × RepairBeamWeapon.java × Weapon.java × WeaponMount.java ×
Project JDK is not defined Setup SDK
37 //lightning bullets need to be initialized first.
38 damageLightning = new BulletType( speed: 0.0001f, damage: 0f){{
39     lifetime = Fx.lightning.lifetime;
40     hitEffect = Fx.hitLancer;
41     despawnEffect = Fx.none;
42     status = StatusEffects.shocked;
43     statusDuration = 10f;
44     hittable = false;
45 }};
46
```

MassDriverBolt.java [link](#)

```
BulletType.java × MassDriverBolt.java × LaserBulletType.java ×
Project JDK is not defined Setup SDK
12
13     public class MassDriverBolt extends BulletType{
14
15         public MassDriverBolt(){
16             super( speed: 1f, damage: 75);
17             collidesTiles = false;
18             lifetime = 1f;
19             despawnEffect = Fx.smeltSmoke;
20             hitEffect = Fx.hitBulletBig;
21         }
22
23         @Override
24         public void draw(Bullet b){
25             float w = 11f, h = 13f;
26
27             Draw.color(Pal.bulletYellowBack);
28             Draw.rect("shell-back", b.x, b.y, w, h, b.rotation() + 90);
29
30             Draw.color(Pal.bulletYellow);
31             Draw.rect("shell", b.x, b.y, w, h, b.rotation() + 90);
32
33             Draw.reset();

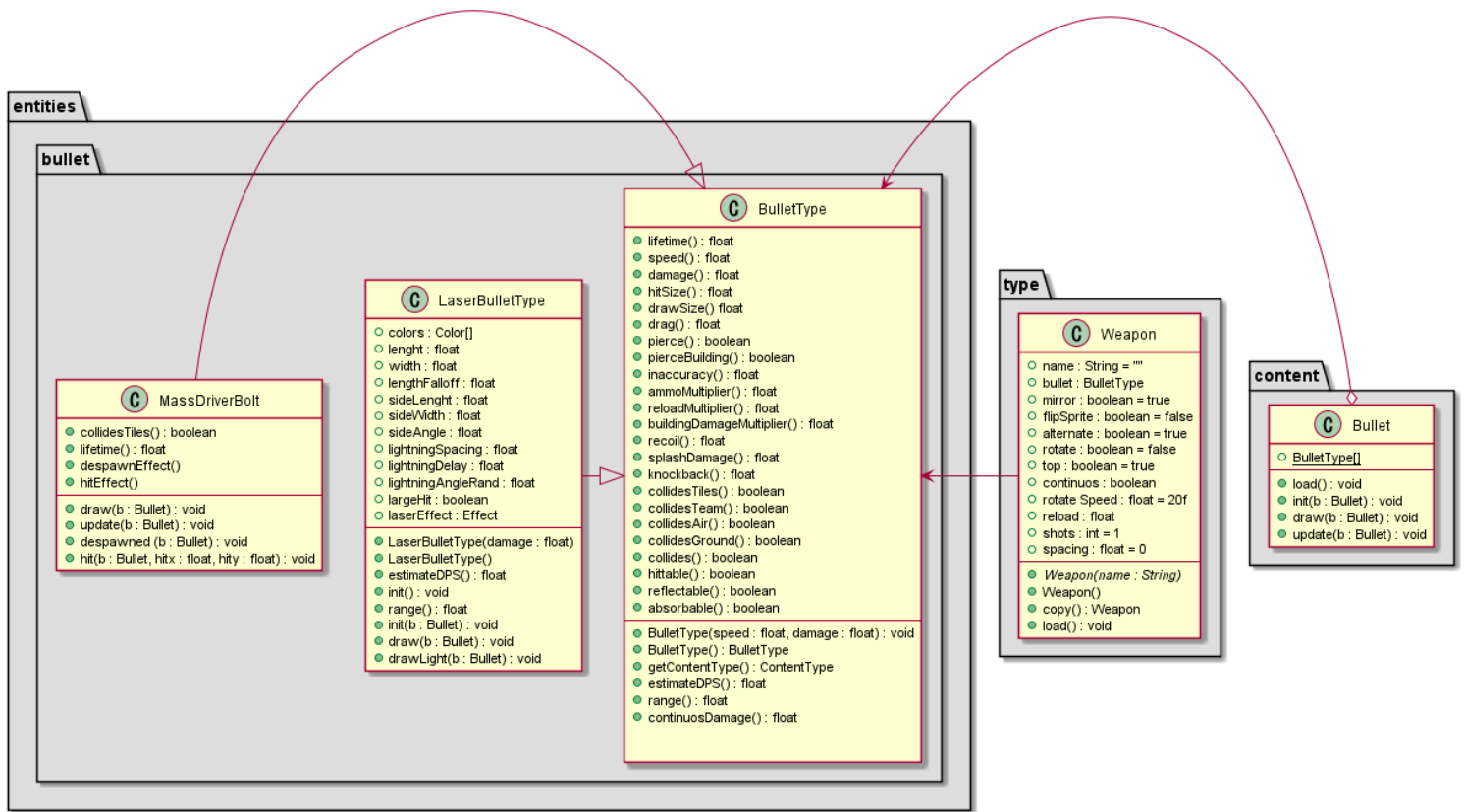
```

LaserBulletType [link](#)



```
11
12 public class LaserBulletType extends BulletType{
13     public Color[] colors = {Pal.lancerLaser.cpy().mul(1f, 1f, 1f, 0.4f), Pal.lancerLaser, Color.white};
14     public Effect laserEffect = Fx.lancerLaserShootSmoke;
15     public float length = 160f;
16     public float width = 15f;
17     public float lengthFalloff = 0.5f;
18     public float sideLength = 29f, sideWidth = 0.7f;
19     public float sideAngle = 90f;
20     public float lightningSpacing = -1, lightningDelay = 0.1f, lightningAngleRand;
21     public boolean largeHit = false;
22
23     public LaserBulletType(float damage){
24         this.damage = damage;
25         this.speed = 0f;
26
27         hitEffect = Fx.hitLaserBlast;
28         hitColor = colors[2];
29         despawnEffect = Fx.none;
30         shootEffect = Fx.hitLancer;
31         smokeEffect = Fx.none;
32         hitSize = 4;
33         lifetime = 16f;
34
35         @Override
36         public void draw(Bullet b){
37             float realLength = b.fdata;
38
39             float f = Mathf.curve(b.fin(), 0f, 0.2f);
40             float baseLen = realLength * f;
41             float cwidth = width;
42             float compound = 1f;
43         }
44     }
45 }
```

UML Class Diagram



4. Update Method

Design Pattern References : [reference1](#) , [reference2](#)

ในเกม Mindustry จะมีการ update ค่าของ Object ต่างๆหลายตัว
ในการทำงานทุกๆเฟรม จึงมีการนำตัว Update Method มาใช้งานเพื่อนช่วยในการ
Update ค่าของ Object ต่างๆหลายตัวนั้น

จากโค้ดตัวอย่างมี abstract class Ability ที่มีการกำหนด method update ไว้
และให้ subclass แต่ละตัวที่นำไปใช้ implement ตัว update ค่าของตัวเอง และมีคลาส
UnitComp ที่เรียกการใช้งาน update ของแต่ละ object ที่สร้างไว้

Source Code

Ability.java [link](#)

```
entities > abilities > Ability.java > Ability > update(Unit)
6
7  public abstract class Ability implements Cloneable{
8      public void update(Unit unit){}
9      public void draw(Unit unit){}
10     public void death(Unit unit){}
11
12     public Ability copy(){
13         try{
14             return (Ability)clone();
15         }catch(CloneNotSupportedException e){
16             //I am disgusted
17             throw new RuntimeException(message: "java sucks", e);
18         }
19     }
20
21     public void displayBars(Unit unit, Table bars){
22
23     }
24
25     /** @return localized ability name; mods should override this. */
26     public String localized(){
27         return Core.bundle.get("ability." + getClass().getSimpleName().replace(target: "Ability", replacement: "").toLowerCase());
28     }
29 }
30
```

EnergyFieldAbility.java [link](#)

```
Ability.java 1 • EnergyFieldAbility.java 4 X DefenderAI.java 2 • Control.java 2 • Conveyor.java 2 •
> abilities > EnergyFieldAbility.java > EnergyFieldAbility
public class EnergyFieldAbility extends Ability{
    private static final Seq<Healthc> all = new Seq<>();

    public float damage = 1, reload = 100, range = 60;
    public Effect healEffect = Fx.heal, hitEffect = Fx.hitLaserBlast, damageEffect = Fx.chainLightning;
    public StatusEffect status = StatusEffects.electrified;
    public Sound shootSound = Sounds.spark;
    public float statusDuration = 60f * 6f;
    public float x, y;
    public boolean targetGround = true, targetAir = true, hitBuildings = true, hitUnits = true;
    public int maxTargets = 25;
    public float healPercent = 2.5f;

    public float layer = Layer.bullet - 0.001f, blinkScl = 20f, blinkSize = 0.1f;
    public float effectRadius = 5f, sectorRad = 0.14f, rotateSpeed = 0.5f;
    public int sectors = 5;
    public Color color = Pal.heal;
    public boolean useAmmo = true;

    protected float timer, curStroke;
    protected boolean anyNearby = false;

    EnergyFieldAbility(){

}

}

@Override
public void update(Unit unit){
    curStroke = Mathf.lerpDelta(curStroke, anyNearby ? 1 : 0, 0.09f);

    if((timer += Time.delta) >= reload && (!useAmmo || unit.ammo > 0 || !state.rules.unitAmmo)){
        Tmp.v1.trns(unit.rotation - 90, x, y).add(unit.x, unit.y);
        float rx = Tmp.v1.x, ry = Tmp.v1.y;
        anyNearby = false;

        all.clear();

        if(hitUnits){
            Units.nearby(team: null, rx, ry, range, other -> {
                if(other != unit && (other.isFlying() ? targetAir : targetGround)){
                    all.add(other);
                }
            });
        }
    }
}
```

ForceFieldAbility.java [link](#)

```
ava 2 ● ● Ability.java 1 ● ForceFieldAbility.java 1 × DefenderAI.java 2 ● Control.java 2 ● Conveyor.jav

entities > abilities > ForceFieldAbility.java > ForceFieldAbility
14 import mindustry.ui.*;
15
16 public class ForceFieldAbility extends Ability{
17     /** Shield radius. */
18     public float radius = 60f;
19     /** Shield regen speed in damage/tick. */
20     public float regen = 0.1f;
21     /** Maximum shield. */
22     public float max = 200f;
23     /** Cooldown after the shield is broken, in ticks. */
24     public float cooldown = 60f * 5;
25
26     /** State. */
27     protected float radiusScale, alpha;
28
29     private static float realRad;
30     private static Unit paramUnit;
31     private static ForceFieldAbility paramField;
32     private static final Cons<Bullet> shieldConsumer = trait -> {

Ability.java 1 ● ForceFieldAbility.java 1 × DefenderAI.java 2 ● Control.java 2 ● Conveyor.java 2 ● World.java 2 ● Map
> abilities > ForceFieldAbility.java > ForceFieldAbility
@Override
public void update(Unit unit){
    if(unit.shield < max){
        unit.shield += Time.delta * regen;
    }

    alpha = Math.max(alpha - Time.delta/10f, 0f);

    if(unit.shield > 0){
        radiusScale = Mathf.lerpDelta(radiusScale, 1f, 0.06f);
        paramUnit = unit;
        paramField = this;
        checkRadius(unit);

        Groups.bullet.intersect(unit.x - realRad, unit.y - realRad, realRad * 2f, realRad * 2f, shieldConsumer);
    }else{
        radiusScale = 0f;
    }
}
```

MoveLightningAbility.java [Link](#)

```

11 import mindustry.gen.*;
12 import mindustry.entities.bullet.*;
13
14 public class MoveLightningAbility extends Ability{
15     /** Lightning damage */
16     public float damage = 35f;
17     /** Chance of firing every tick. Set >= 1 to always fire lightning every tick at max s
18     public float chance = 0.15f;
19     /** Length of the lightning. <= 0 to disable */
20     public int length = 12;
21     /** Speeds for when to start lightninging and when to stop getting faster */
22     public float minSpeed = 0.8f, maxSpeed = 1.2f;
23     /** Lightning color */
24     public Color color = Color.valueOf("a9d8ff");
25     /** Shifts where the lightning spawns along the Y axis */
26     public float offset = 0f;
27     /** Offset along the X axis */
28     public float width = 0f;
29     /** Whether the spawn side alternates */
30     public boolean alternate = true;
31     /** Jittering heat sprite like the shield on v5 Javelin */
32     public String heatRegion = "error";
33     /** Bullet type that is fired. Can be null */
34     public @Nullable BulletType bullet;
35     /** Bullet angle parameters */
36     public float bulletAngle = 0f, bulletSpread = 0f;
37
38     public Effect shootEffect = Fx.sparkShoot;
39     public boolean parentizeEffects;
40     public Sound shootSound = Sounds.spark;
41

```

```

67 @Override
68 public void update(Unit unit){
69     float scl = Mathf.clamp((unit.vel().len() - minSpeed) / (maxSpeed - minSpeed));
70     if(Mathf.chance(Time.delta * chance * scl)){
71         float x = unit.x + Angles.trnsx(unit.rotation, offset, width * side), y = unit.y + Angles.trnsy(unit.rotation, offset,
72
73         shootEffect.at(x, y, unit.rotation, color, parentizeEffects ? unit : null);
74         shootSound.at(x, y);
75
76         if(length > 0){
77             Lightning.create(unit.team, color, damage, x + unit.vel.x, y + unit.vel.y, unit.rotation, length);
78         }
79
80         if(bullet != null){
81             bullet.create(unit, unit.team, x, y, unit.rotation + bulletAngle + Mathf.range(bulletSpread));
82         }
83
84         if(alternate) side *= -1f;
85     }
86 }
87

```


UnitComp.java [link](#) (class ที่เรียกใช้ update)

```
UnitComp.java 5 • Ability.java 1 • DefenderAI.java 2 • Control.java 2 • Conveyor.java 2 • World.java 2 •
ies > comp > UnitComp.java > UnitComp > abilities
abstract class UnitComp implements Healthc, Physicsc, Hitboxc, Statusc, Teamc, Itemsc, Rotc, Unitc, Weaponsc, I

    @Import boolean hovering, dead, disarmed;
    @Import float x, y, rotation, elevation, maxHealth, drag, armor, hitSize, health, ammo, minFormationSpeed,
    @Import Team team;
    @Import int id;
    @Import @Nullable Tile mineTile;
    @Import Vec2 vel;
    @Import WeaponMount[] mounts;

    private UnitController controller;
    UnitType type = UnitTypes.alpha;
    boolean spawnedByCore;
    double flag;

    transient float shadowAlpha = -1f;
    transient Seq<Ability> abilities = new Seq<>(0);
    transient float healTime;
    private transient float resupplyTime = Mathf.random(10f);
    private transient boolean wasPlayer;
    private transient boolean wasHealed;

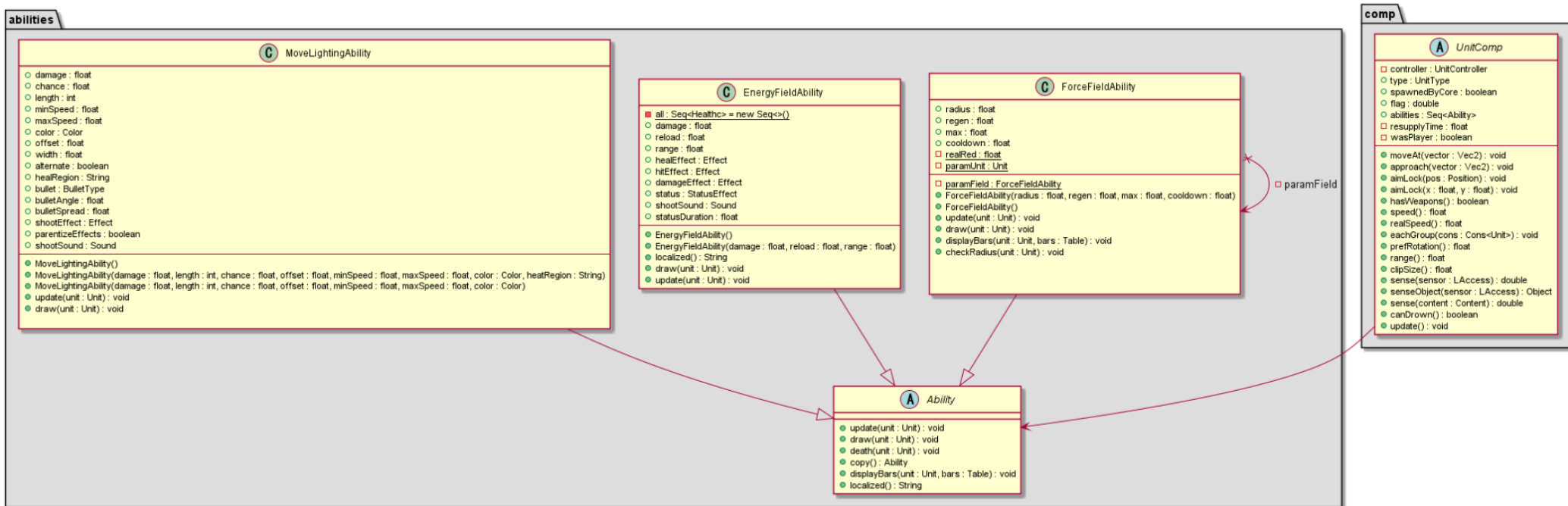
    /** Called when this unit was unloaded from a factory or spawn point. */
    public void unloaded(){

    }
}
```

```
UnitComp.java 5 • Ability.java 1 • DefenderAI.java 2 • Control.java 2 •
ies > comp > UnitComp.java > UnitComp > update()
    }

    if(abilities.size > 0){
        for(Ability a : abilities){
            a.update(self());
        }
    }
}
```

UML Class Diagram



แหล่งอ้างอิง

1. [Servers - Mindustry](#)
2. [Dedicated Server Architecture](#)
3. [How to Configure Autosaving of Your Minecraft World - Knowledgebase - Shockbyte](#)
4. [Getting Started - Mindustry Documentation \(mindustrymodders.gitlab.io\)](#)
5. [Frequently Asked Questions - Mindustry Wiki \(mindustrygame.github.io\)](#)
6. [Scripting - Mindustry Wiki \(mindustrygame.github.io\)](#)
7. [Update Method : Game Programming PatternsSequencing Patterns](#)
8. [Update Method : java-design-patterns.com](#)