

# Minecraft の MOD で RPG ゲームを構築する。



2023 年度プロジェクトリサーチ

# Minecraft の MOD で RPG ゲームを構築する

## 活動報告書

メンバー : Y210156 竹内慎作  
Y210164 寺田倫太郎  
Y210169 長谷川龍弥  
Y210170 林大輔  
Y210177 福田隼己

## 目的

このグループでは、RPG ゲームの構築を通して RPG ゲームの仕組みやチームで開発する大変さを学ぶ。

## 計画

活動は次の手順で進めていく。まず、モンスターやアイテム、武器などを1つだけ追加する。情報共有のための資料を作る。そしてここからは、作業分担で次のことを分担しながら行っていく。まず、モンスターやアイテム、武器の数を増やす作業。次に、ステータスを追加する作業。最後にRPGのマップを作る作業。大きく分けてこれらのことを作業分担した。その後、これらの要素を追加することができた後作ったRPGマップにダンジョンや建築物を全員で追加していきました。

次にこれらの計画の主な役割や目的について説明していく。まず、モンスターやアイテム、武器などを1つだけ追加する作業と情報共有の資料を作る作業、これらの目的について説明していく。これは、一度モンスターやアイテムを追加しそれについての資料を作成することでほかのメンバーが新たにモンスターやアイテムを追加する際にコードや情報共有用の資料を見ることで容易に追加することができるようになる。

次にモンスターやアイテム、武器を追加する作業。この作業の目的は、モンスターや武器、アイテムを追加することでゲームの内容を濃くすることが目的だ。

次にステータスを追加する作業、これはRPGゲームの核ともいえる部分でこの要素がないとRPGゲームとは言えない。

次にRPGマップの作成する作業。この作業の目標はRPGゲームの舞台となるマップの作成だ、このマップの製作には専用のソフトを使い作成していった。詳しくは、活動経過のマップについてで、説明する。

最後にRPGマップにダンジョンや建築物を追加する作業である。この作業の目的は、学園祭の発表の際に来て下さったお客様にどのような活動を行ってきたか分かりやすくするのが主な目的だ。

次にメンバーのそれぞれがどのような活動をしてきたかをまとめる。

Y210156 竹内禎作

シナリオの作成

RPG マップ内でのダンジョンの作成や建築物の追加

RPG マップの作成

Y210164 寺田倫太郎

モンスターや武器の追加

RPG マップ内でのダンジョンの作成や建築物の追加

Y210169 長谷川龍弥

モンスターの追加

RPG マップ内でのダンジョンの作成や建築物の追加

Y210170 林大輔

モンスターや武器の追加

RPG マップ内でのダンジョンの作成や建築物の追加

RPG ゲームサーバーの立ち上げ

プレイヤーのセーブポイントやリスポーン地点の作成

Y210177 福田隼己

モンスターやアイテムの追加

情報共有用の資料の作成

RPG マップ内でのダンジョンの作成や建築物の追加

ステータスの追加

## 調査方法

まず、RPG ゲームの仕組みを理解するためにステータスを追加する際にどのような計算の仕方をすればステータスを追加できるかを考える。次に、モンスターやアイテム、武器を追加する過程でどのパラメータが攻撃力や HP を構築しているか、アイテムにどのような構文を追加すれば武器として認識させることができるか、アイテムが消失しないような特性を与えることができるかなどを考える。

次にチームで開発する大変さを学ぶために、様々な方法で情報共有の方法を図り、どの方法が一番情報の伝達に向いているかを考える。

## 活動経過

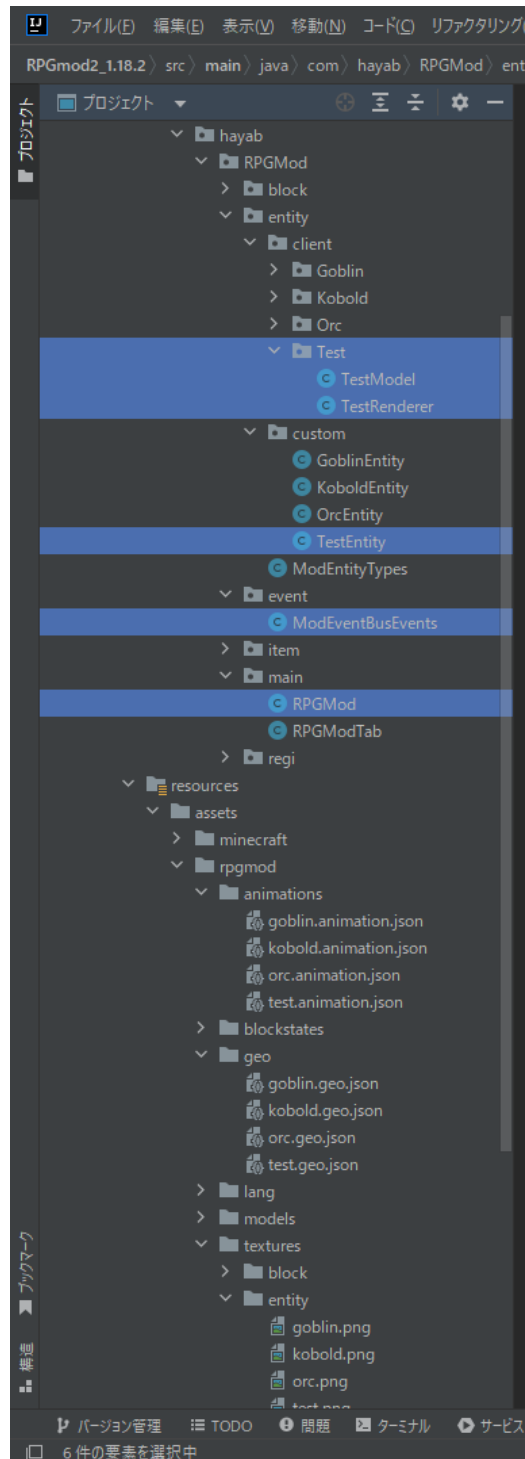
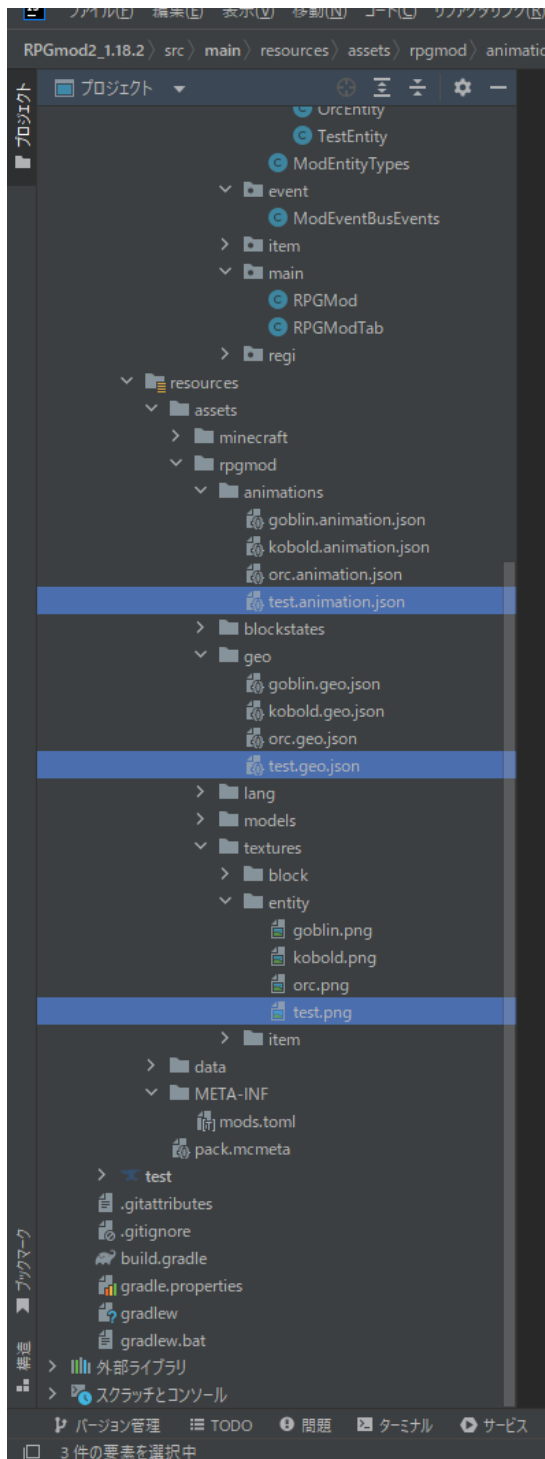
ここから、主な活動経過をまとめていく。

1. モンスター
2. アイテム、武器
3. ステータス
4. RPG マップ(ダンジョンや建築物)

# 1. モンスター

## 1.1 必要なファイル

以下の画像にモンスターの製作に当たり必要なファイルの画像を貼り付ける

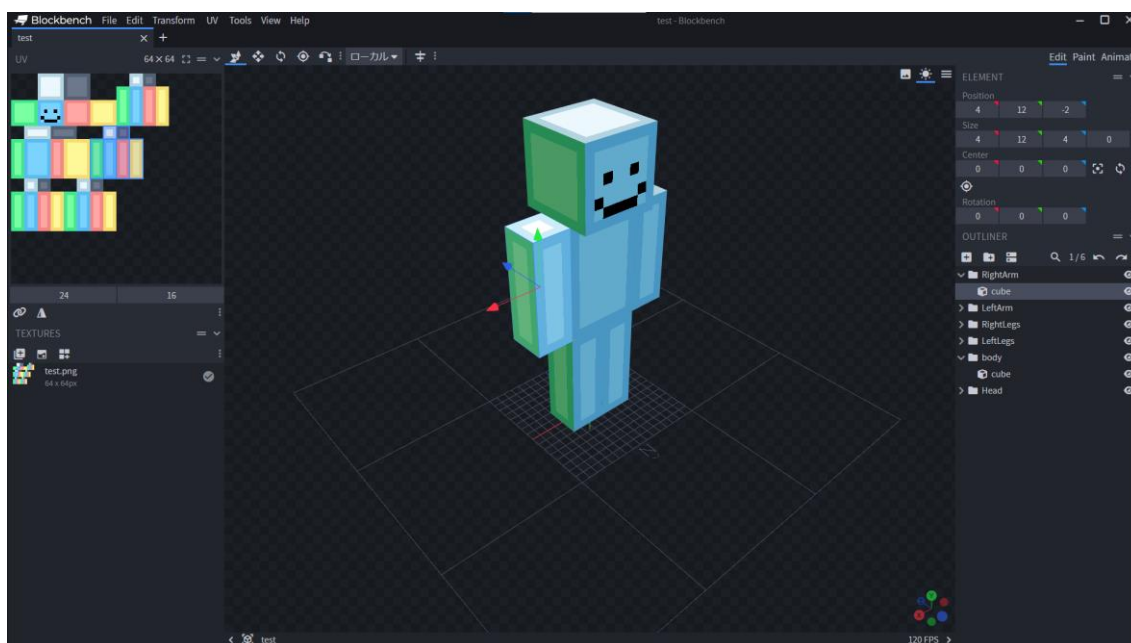


まず、モンスターを追加する際は前のページに乗せたファイルがいる。ここからはそれぞれのファイルで主にどのようなことをしているか解説する。まず、左側のジェイソンファイルやピングファイルについて説明する。

## 1.2 モンスターのテクスチャやモーショ

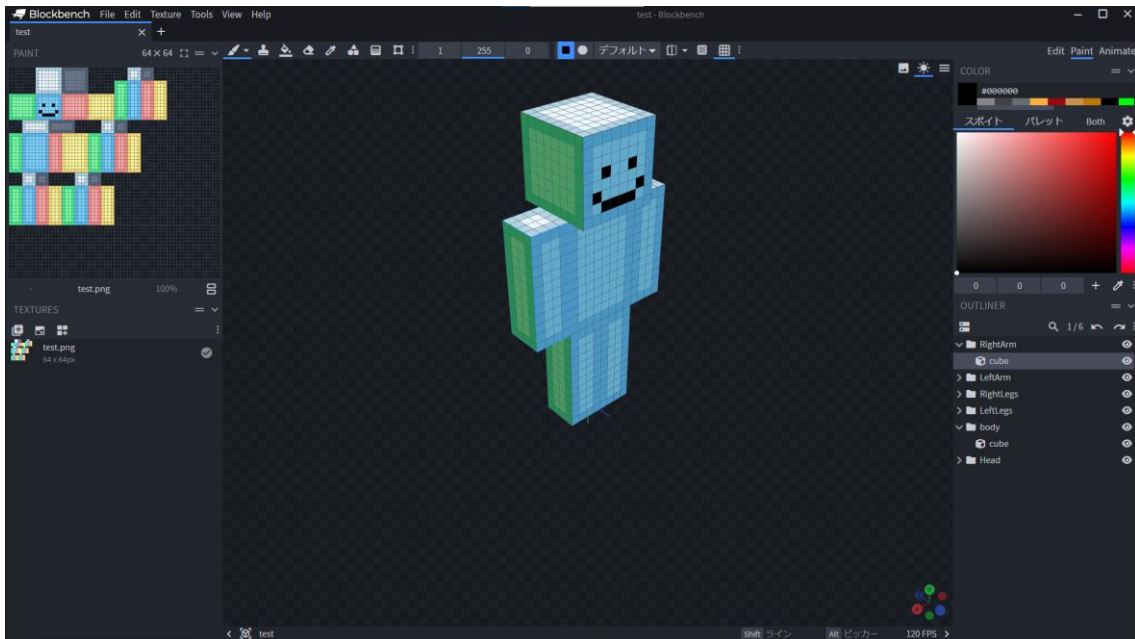
まず、これら 3 つのファイルを作るには、ブロックベンチというソフトを使いモンスターの外見やどのような動きをするかを作る必要がある。ブロックベンチでどのような作業をしたか簡単に画像を使いながら解説する。

### ・キューブの設置



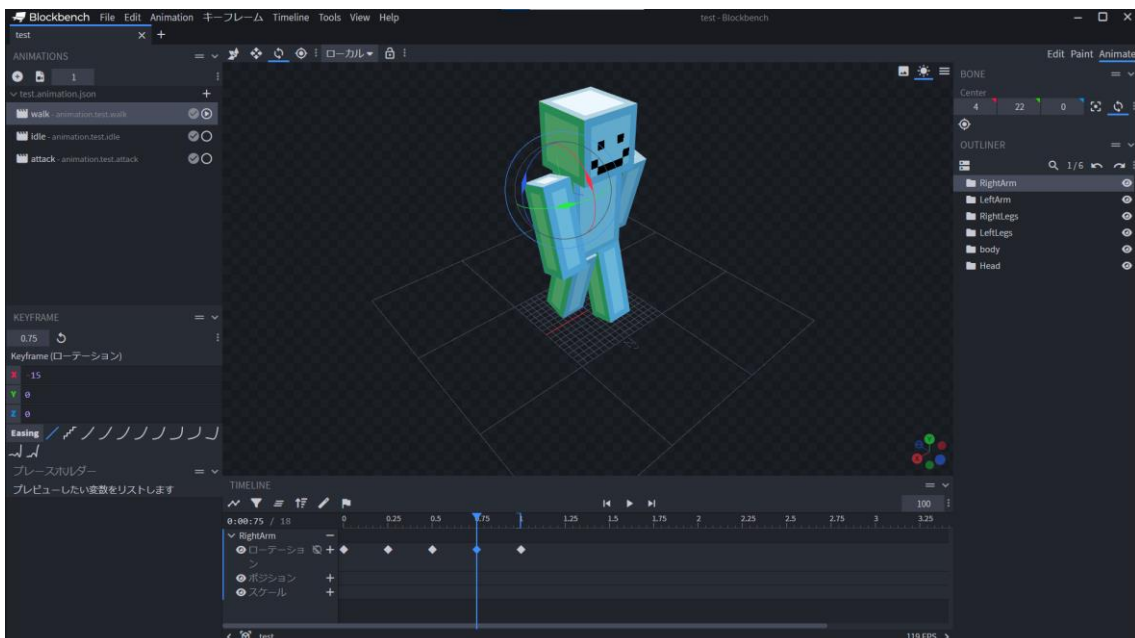
まず、この画像では腕や胴体、顔などキューブを設置し、大きさを変えることで外見を作成している。

## ・色を付ける



この画像では先ほど作ったキューブに右側のパレットから色を選び塗っていくことができる。

## ・アニメーションの作成



この画像では、関節を1つ1つ動かしてコマ送りの要領で今まで作ってきたモンスターの動きを作っていく。



## ・ファイルの出力

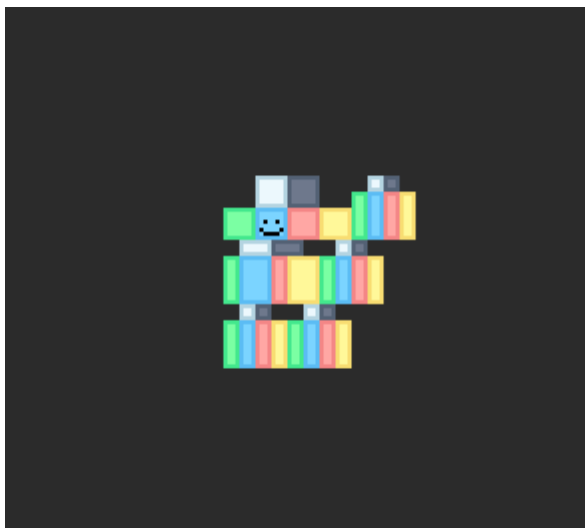
ここまでで作ったファイルをそれぞれ出力する。出力すると下のようなファイルが 3 つできる。

```
1 {
2   "format_version": "1.8.0",
3   "animations": {
4     "animation.test.walk": {
5       "loop": true,
6       "animation_length": 1,
7       "bones": {
8         "RightArm": {
9           "rotation": {
10            "0.0": {
11              "vector": [0, 0, 0]
12            },
13            "0.25": {
14              "vector": [15, 0, 0]
15            },
16            "0.5": {
17              "vector": [0, 0, 0]
18            },
19            "0.75": {
20              "vector": [-15, 0, 0]
21            },
22            "1.0": {
23              "vector": [0, 0, 0]
24            }
25          }
26        },
27        "LeftArm": {
28          "rotation": {
29            "0.0": {
30              "vector": [0, 0, 0]
31            },
32            "0.25": {
33              "vector": [-15, 0, 0]
34            },
35            "0.5": {
36              "vector": [0, 0, 0]
37            },
38            "0.75": {
39              "vector": [15, 0, 0]
40            },
41            "1.0": {
42              "vector": [0, 0, 0]
43            }
44          }
45        },
46        "RightLegs": {
47          "rotation": {
48            "0.0": {
49              "vector": [0, 0, 0]
```

test.animation.json

```
1 {
2   "format_version": "1.12.0",
3   "minecraft:geometry": [
4     {
5       "description": {
6         "identifier": "geometry.unknown",
7         "texture_width": 64,
8         "texture_height": 64,
9         "visible_bounds_width": 2,
10        "visible_bounds_height": 3.5,
11        "visible_bounds_offset": [0, 1.25, 0]
12      },
13      "bones": [
14        {
15          "name": "RightArm",
16          "pivot": [-4, 22, 0],
17          "cubes": [
18            { "origin": [-8, 12, -2], "size": [4, 12, 4], "uv": [24, 16] }
19          ]
20        },
21        {
22          "name": "LeftArm",
23          "pivot": [4, 22, 0],
24          "cubes": [
25            { "origin": [4, 12, -2], "size": [4, 12, 4], "uv": [0, 32] }
26          ]
27        },
28        {
29          "name": "RightLegs",
30          "pivot": [0, 12, 0],
31          "cubes": [
32            { "origin": [-4, 0, -2], "size": [4, 12, 4], "uv": [16, 32] }
33          ]
34        },
35        {
36          "name": "LeftLegs",
37          "pivot": [0, 12, 0],
38          "cubes": [
39            { "origin": [0, 0, -2], "size": [4, 12, 4], "uv": [32, 0] }
40          ]
41        },
42        {
43          "name": "body",
44          "pivot": [0, 0, 0],
45          "cubes": [
46            { "origin": [-4, 12, -2], "size": [8, 12, 4], "uv": [0, 16] }
47          ]
48        }
49      ]
50    }
51  ]
52 }
```

test.geo.json



test.png

## ・ファイルの解説

### **test.animation.json**

このJSONファイルは、どんな動きをするのか書かれており移動していないとき、歩いているとき、攻撃するときそれぞれの動きが書かれている。

### **test.geo.json**

このJSONファイルは、どこの位置に体を構成するキューブが設置されているか、どのくらいの大きさかが書かれている。

### **test.png**

このPNGファイルはモンスターのテクスチャである。



## • ModEntityTypes

```
ModEntityTypes.java
package com.hayab.RPGMod.entity;

import ...

1件の使用箇所
public class ModEntityTypes {
    1件の使用箇所
    public static final DeferredRegister<EntityType<?>> ENTITY_TYPES =
        DeferredRegister.create(ForgeRegistries.ENTITIES, RPGMod.MOD_ID);

    2件の使用箇所
    public static final RegistryObject<EntityType<GoblinEntity>> GOBLIN =
        ENTITY_TYPES.register("goblin",
            () -> EntityType.Builder.of(GoblinEntity::new, MobCategory.MONSTER)
                .sized(0.3000f, 0.8f, 0.2000f, 1.05f)
                .build(new ResourceLocation(RPGMod.MOD_ID, "goblin").toString()));

    2件の使用箇所
    public static final RegistryObject<EntityType<KoboldEntity>> KOBOLD =
        ENTITY_TYPES.register("kobold",
            () -> EntityType.Builder.of(KoboldEntity::new, MobCategory.MONSTER)
                .sized(0.3000f, 0.8f, 0.2000f, 2f)
                .build(new ResourceLocation(RPGMod.MOD_ID, "kobold").toString()));

    2件の使用箇所
    public static final RegistryObject<EntityType<OrcEntity>> ORC =
        ENTITY_TYPES.register("orc",
            () -> EntityType.Builder.of(OrcEntity::new, MobCategory.MONSTER)
                .sized(0.3000f, 1f, 0.2000f, 2f)
                .build(new ResourceLocation(RPGMod.MOD_ID, "orc").toString()));

    2件の使用箇所
    public static final RegistryObject<EntityType<TestEntity>> TEST =
        ENTITY_TYPES.register("test",
            () -> EntityType.Builder.of(TestEntity::new, MobCategory.MONSTER)
                .sized(0.3000f, 1f, 0.2000f, 2f)
                .build(new ResourceLocation(RPGMod.MOD_ID, "test").toString()));

    1件の使用箇所
    public static void register(IEventBus eventBus) { ENTITY_TYPES.register(eventBus); }
}
```

このファイルはモンスターのタイプやモンスターの当たり判定を設定している。今回作っているモンスターのタイプは MONSTE という分類に入れている。この MONSTE は Minecraft 内ではプレイヤーを攻撃するモンスターがまとめられているところです。まず、当たり判定とは実際に目に見えるモンスターの大きさとは別に設定されているものです。例えば、実際に見えているサイズより小さく設定して今うとカーソルはモンスターに向いているのに攻撃が当たらないということが起こって今います。そのためこの当たり判定は、実際にゲーム内で当たり判定を表示させながら大きさを変えていきます。

## • ModEventBusEvents

```
ModEventBusEvents.java
1 package com.hayab.RPGMod.event;
2
3 import ...
12
13 1件の使用箇所
14 @Mod.EventBusSubscriber(modid = RPGMod.MOD_ID, bus = Mod.EventBusSubscriber.Bus.MOD)
15 public class ModEventBusEvents {
16     @SubscribeEvent
17     public static void entityAttributeEvent(EntityAttributeCreationEvent event) {
18         event.put(ModEntityTypes.GOBLIN.get(), GoblinEntity.setAttributes());
19         event.put(ModEntityTypes.KOBOLD.get(), KoboldEntity.setAttributes());
20         event.put(ModEntityTypes.ORC.get(), OrcEntity.setAttributes());
21         event.put(ModEntityTypes.TEST.get(), TestEntity.setAttributes());
22     }
23 }
```

このファイルは、Mod.EventBusSubscriber というファイルにここまで作った2つのファイルを紐づけする。

## • TestModel.java

```
TestModel.java
1 package com.hayab.RPGMod.entity.client.Test;
2
3 import com.hayab.RPGMod.entity.custom.TestEntity;
4 import com.hayab.RPGMod.main.RPGMod;
5 import net.minecraft.resources.ResourceLocation;
6 import software.bernie.geckolib3.model.AnimatedGeoModel;
7
8 1件の使用箇所
9 public class TestModel extends AnimatedGeoModel<TestEntity> {
10     @Override
11     public ResourceLocation getModelLocation(TestEntity object) {
12         return new ResourceLocation(RPGMod.MOD_ID, p_135812_ "geo/test.geo.json");
13     }
14
15     @Override
16     public ResourceLocation getTextureLocation(TestEntity object) {
17         return new ResourceLocation(RPGMod.MOD_ID, p_135812_ "textures/entity/test.png");
18     }
19
20     @Override
21     public ResourceLocation getAnimationFileLocation(TestEntity animatable) {
22         return new ResourceLocation(RPGMod.MOD_ID, p_135812_ "animations/test.animation.json");
23     }
24 }
```

このファイルは最初に作った3つのファイルの紐づけを設定している。

## • TestRenderer.java

```
TestRenderer.java
1 package com.hayab.RPGMod.entity.client.Test;
2
3 import ..
4
5 2件の使用箇所
6
7 1件の使用箇所
8 public class TestRenderer extends GeoEntityRenderer<TestEntity> {
9     1件の使用箇所
10     public TestRenderer(EntityRendererProvider.Context renderManager) {
11         super(renderManager, new TestModel());
12         this.shadowRadius = 0.3f;
13     }
14
15     @Override
16     public ResourceLocation getTextureLocation(TestEntity instance) {
17         return new ResourceLocation(RPGMod.MOD_ID, p_135812_ "textures/entity/test.png");
18     }
19
20     @Override
21     public RenderType getRenderType(TestEntity animatable, float partialTick, PoseStack stack,
22                                     @Nullable MultiBufferSource bufferSource,
23                                     @Nullable VertexConsumer buffer, int packedLight,
24                                     ResourceLocation texture) {
25         stack.scale(p_85842_ 1.0f, p_85843_ 1.0f, p_85844_ 1.0f);
26         return super.getRenderType(animatable, partialTick, stack, bufferSource, buffer, packedLight, texture);
27     }
28 }
```

このファイルはモンスターの実際に目に見えるサイズを設定している

## • RPGMod.java

```
1 package com.hayab.RPGMod.main;
2
3 import ...;
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18 @Mod("rpgmod")
19 public class RPGMod {
20
21     public static final String MOD_ID = "rpgmod";
22
23     2件の使用箇所
24     public static final CreativeModeTab RPGMOD_TAB = new RPGModTab();
25
26     public RPGMod() {
27         IEventBus modEventBus = FMLJavaModLoadingContext.get().getModEventBus();
28         ModEntityTypes.register(modEventBus);
29
30         GeckoLib.initialize();
31     }
32
33     1件の使用箇所
34     @Mod.EventBusSubscriber(modid = MOD_ID, bus = Mod.EventBusSubscriber.Bus.MOD)
35     public static class ClientModEvent {
36         @SubscribeEvent
37         public static void onClientSetup(FMLClientSetupEvent event){
38             EntityRenderers.register(ModEntityTypes.GOBBLIN.get(), GoblinRenderer::new);
39             EntityRenderers.register(ModEntityTypes.KOBOLD.get(), KoboldRenderer::new);
40             EntityRenderers.register(ModEntityTypes.ORB.get(), OrcRenderer::new);
41             EntityRenderers.register(ModEntityTypes.TEST.get(), TestRenderer::new);
42         }
43     }
44 }
```

このファイルは、ここまでにしたすべてのファイルを MOD に紐づけしてゲーム内に出力できるようにしている。

### 1.4 作成したモンスター

ここからはだれがどのようなモンスターを作成したか説明していきます。

#### Y210164 寺田倫太郎

黒騎士

紫騎士

炎男

ゴーレム

ロボット(黒)

ロボット(黄)

サボテン

雪男

ワーム

#### Y210169 長谷川龍弥

山賊の手下

山賊のボス

Y210170 林大輔

ジュウ君

スライム

ゾンビ

Y210177 福田隼己

ゴブリン

コボルト

オーク

### 1.5 参考資料・使用ソフト

ここからはモンスターを追加する際に使用したソフトや資料をまとめます。

・参考動画

[MODELLING A NEW ENTITY | Blockbench w/ GeckoLib #1 - YouTube](#)

・前提 MOD

Forge : [Downloads for Minecraft Forge for Minecraft 1.18.2](#)

GeckoLib: [GeckoLib - Minecraft Mods - CurseForge](#)

・使用ソフト

IntelliJ IDEA : [IntelliJ IDEA - Java と Kotlin の最先端 IDE \(jetbrains.com\)](#)

ブロックベンチ : [Blockbench](#)

## 2 アイテム・武器

### 2.1 アイテムを作成する際に必要なファイル

まずアイテムを作成に必要なファイルをまとめる。

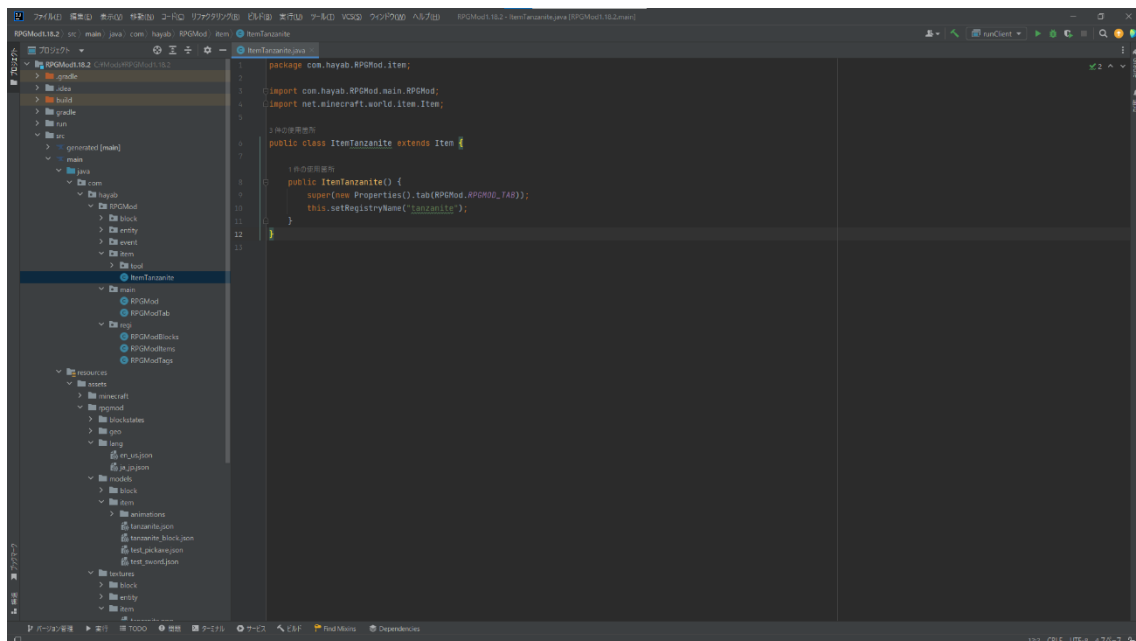
1. ItemTanzanite.java
2. RPGModItems.java
3. Tanzanite.json
4. en\_us.json
5. ja\_jp.json
6. Tanzanite.png

基本的に上のファイルでアイテムは構成されている。

### 2.2 ファイルの説明

ここからはそれぞれのファイルの役割を解説していく。

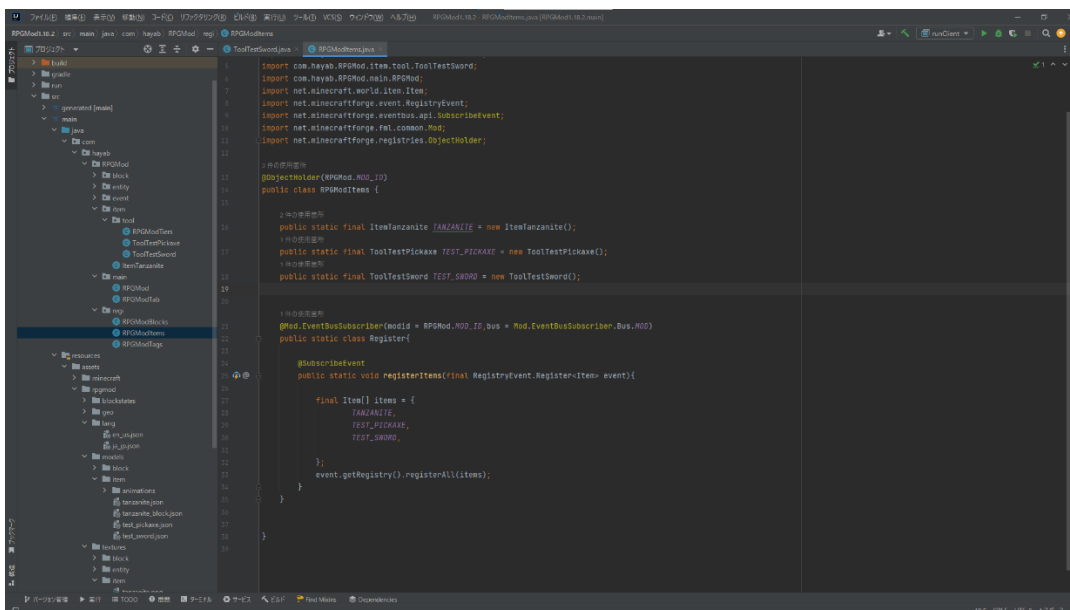
#### ・ ItemTanzanite.java



このファイルはアイテムを指定する引数などを書いてあります。ここで書かれた名前は他のファイルでアイテムを指定する際によく使います。



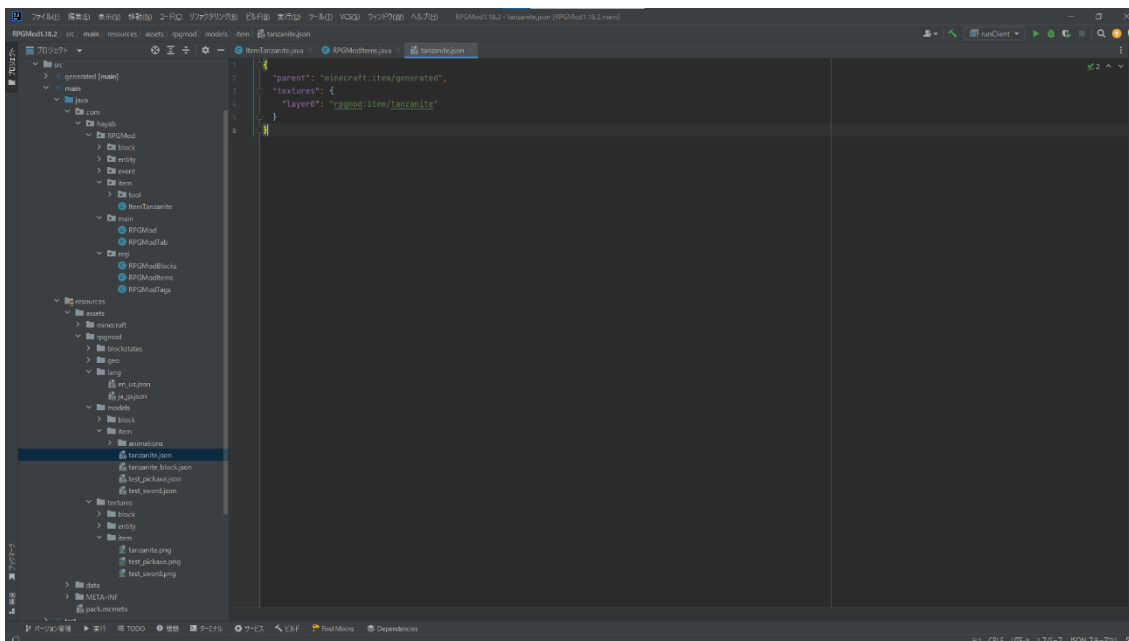
## ・RPGModItems.java



```
1 import com.hayab.RPGMod.item.ToolTestSword;
2 import com.hayab.RPGMod.item.ToolTestPickaxe;
3 import net.minecraft.item.Item;
4 import net.minecraftforge.event.RegistryEvent;
5 import net.minecraftforge.eventbus.api.SubscribeEvent;
6 import net.minecraftforge.fml.common.Mod;
7 import net.minecraftforge.registries.ObjectHolder;
8
9 @ObjectHolder(RPGMod.MOD_ID)
10 public class RPGModItems {
11
12     2 静的変数宣言
13     public static final ItemTanzanite TANZANITE = new ItemTanzanite();
14     3 静的変数宣言
15     public static final ToolTestPickaxe TEST_PICKAXE = new ToolTestPickaxe();
16     4 静的変数宣言
17     public static final ToolTestSword TEST_SWORD = new ToolTestSword();
18
19     5 静的メソッド宣言
20     @Mod.EventBusSubscriber(modid = RPGMod.MOD_ID, bus = Mod.EventBusSubscriber.Bus.MOD)
21     public static class Register{
22
23         @SubscribeEvent
24         public static void registerItems(final RegistryEvent.Register<Item> event){
25
26             final Item[] items = {
27                 TANZANITE,
28                 TEST_PICKAXE,
29                 TEST_SWORD
30             };
31             event.getRegistry().registerAll(items);
32         }
33     }
34 }
```

このファイルでは先ほど作った引数がアイテムであることを決定している。基本的に作られたアイテムや武器はここにまとめられておりどんなアイテムの一覧を見たい場合はこのファイルを見ればわかる。

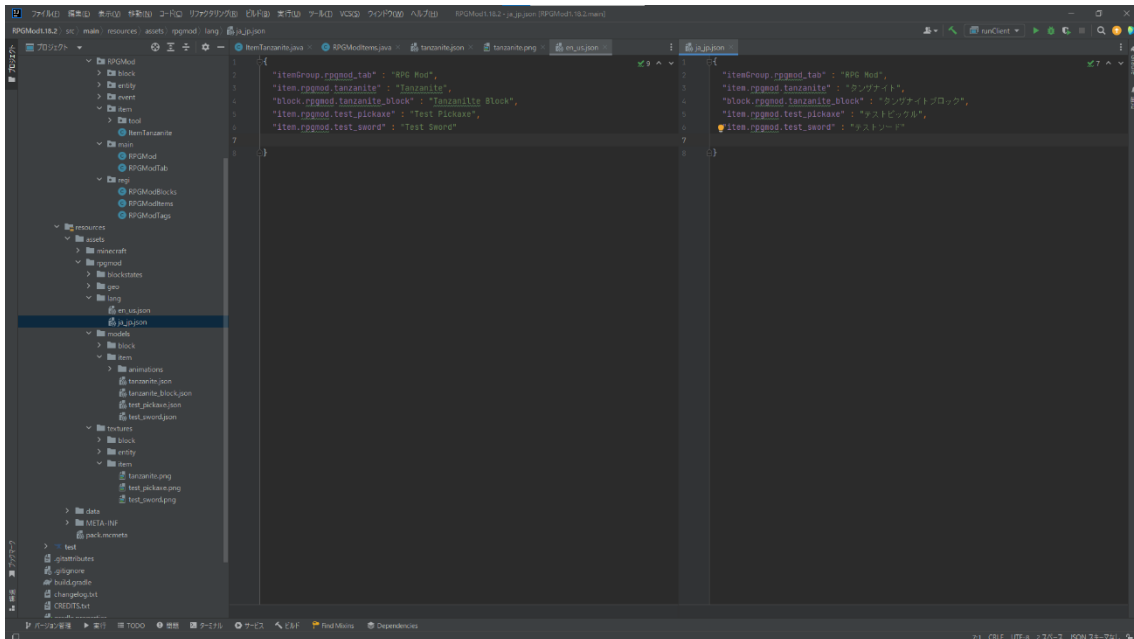
## ・Tanzanite.json



```
1 {
2   "parent": "minecraft:item/generated",
3   "textures": {
4     "layer0": "rpgmod:item/tanzanite"
5   }
6 }
```

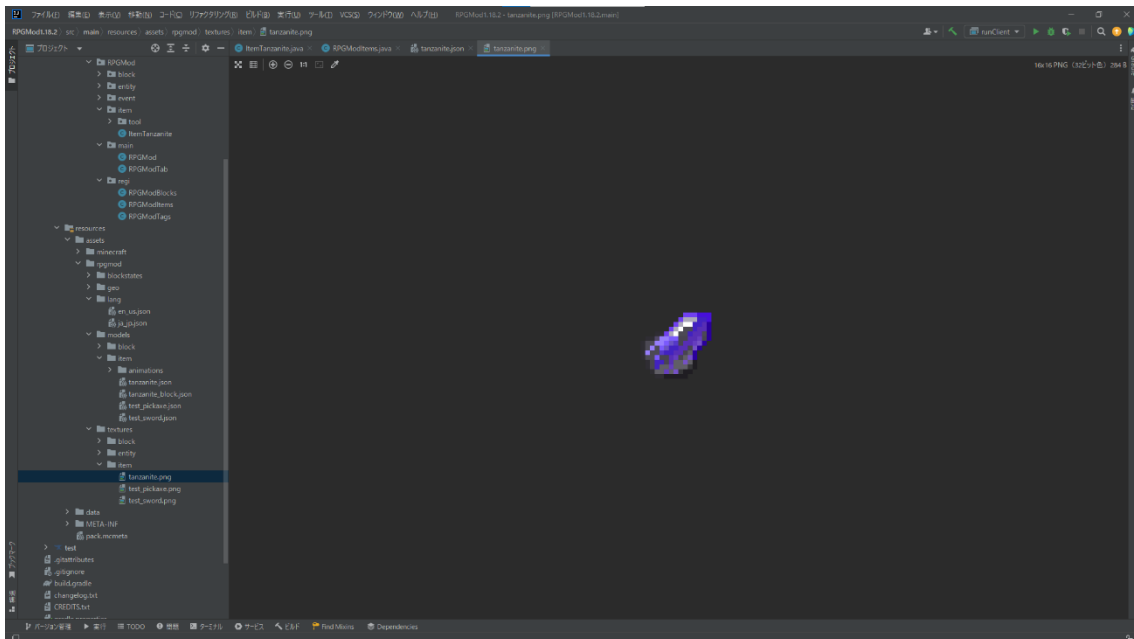
このファイルではアイテムがゲーム内でどのくりにされるかを定める。今回だとrpgmodというくりに入れている。

## • en\_us.json,ja\_jp.json



この二つのファイルは追加したいアイテムの日本語表示名と英語表示名を決めている。ここでは打ち込んだ文字がそのままゲーム内で表示される名前になる。

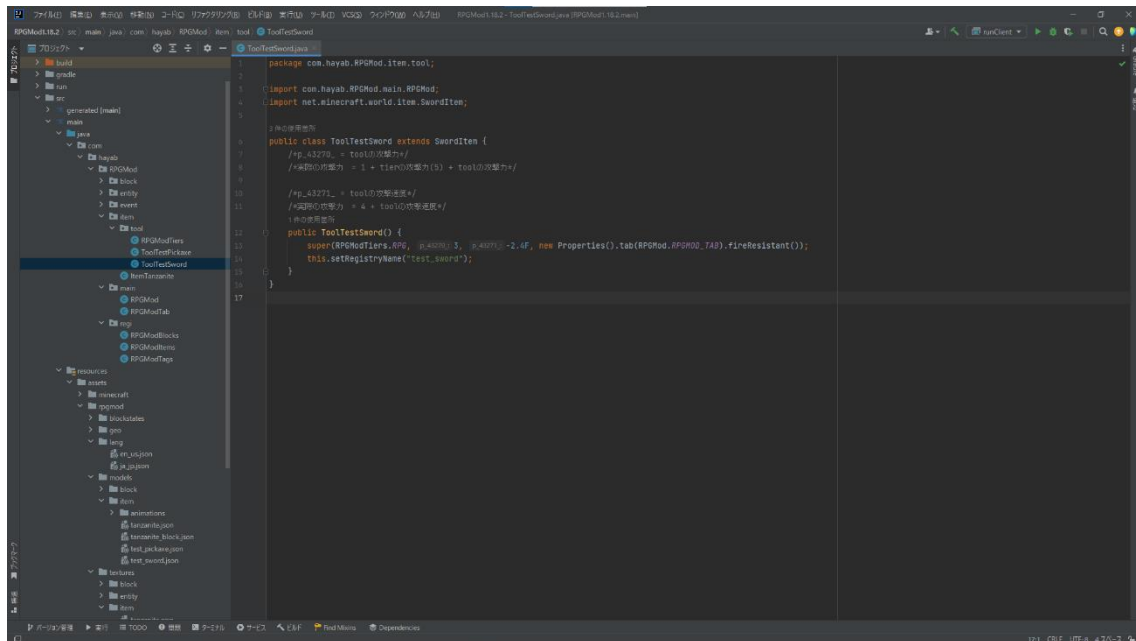
## • tanzanite.png



このファイルはアイテムをどのように表示させるかというものだ。この画像を作る際は mcreator というソフトを用いて作った。

## 2.3 武器の追加

武器を追加する際は上記で説明したファイルのうち ItemTanzanite.java というファイルの内容を書き換える必要がある



```
1 package com.hayab.RFGMod.item.tool;
2
3 import com.hayab.RFGMod.main.RFGMod;
4 import net.minecraft.world.item.SwordItem;
5
6 // 独自の攻撃力
7 public class ToolTestSword extends SwordItem {
8     //pg_43279_ = toolの攻撃力/
9     //床際の攻撃力 = 1 + (tier)の攻撃力(5) + toolの攻撃力/
10
11     //pg_43271_ = toolの攻撃速度*/
12     //床際の攻撃速度 = 4 + toolの攻撃速度*/
13     // 独自の属性
14     public ToolTestSword() {
15         super(RFGModTiers.RPG, 0, 43271, 2, 4F, new Properties(), tab(RFGMod.RFGMOD_TAB).fireResistant());
16         this.setRegistryName("test_sword");
17     }
18 }
```

このファイルでは先ほどはなかった武器の攻撃力やこうげき速度などを追加している。

## 2.4 参考資料・使用ソフト

参考動画

アイテムの追加の仕方：[【自作 Mod の作り方】『アイテムの追加①』マイクラ 1.18.1 \(日本語解説\) part.4 【Minecraft Modding】 - YouTube](#)

武器の追加の仕方：[【自作 Mod の作り方】『ツールの追加①&タグの生成』マイクラ 1.18.1 \(日本語解説\) part.8 【Minecraft Modding】 - YouTube](#)

使用ソフト

Mcreator：[Download MCreator | MCreator](#)

### 3 ステータス

#### 3.1 レベルアップ

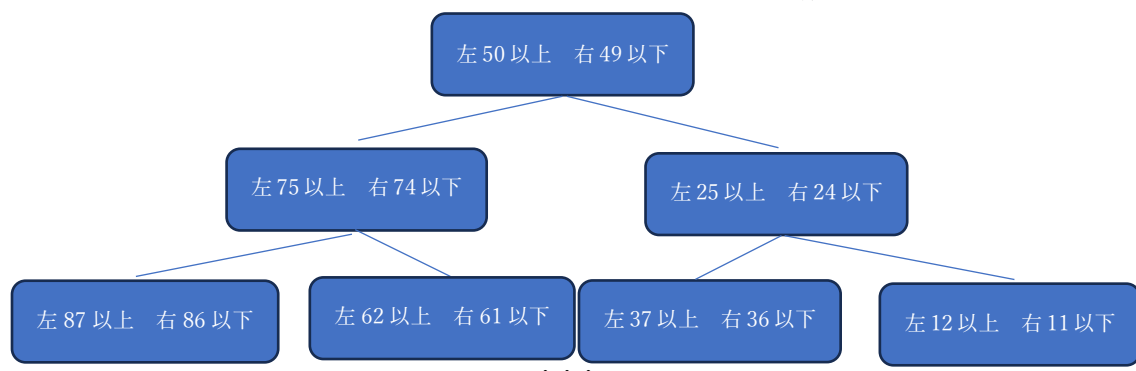
まず、レベルが上がる仕組みについて説明していきます。プレイヤーには必要な経験値量と現在持っている経験値の量、そしてレベルの引数を持っています。もし必要な経験値が現在の経験値量以上となったときに現在の経験値量から必要な経験値を引き、レベルを1あげて必要経験値量の数値を1.1倍にします。そして、もし現在の経験値量がまだ必要経験値より多かった場合もう一度同じ処理をして、現在の経験値量より必要な経験値量の方が大きくなるまで繰り返します。

#### 3.2 ステータス

次にステータスが上がる仕組みについて説明していきます。まず、レベルアップの処理がすべて終わった後にHPや攻撃力などの数値をあげる処理をする関数を順番に処理していきます。

この関数では現在のレベルに対応するステータスの数値を代入するという形で処理をしています。そして、Minecraft内のコードでは1行で完結する処理しか行えないのでレベルに対応するステータスの数値を1行ごとに処理していく必要があります。今回レベルの最大値は100で作っています。つまり、一つのステータスの値を処理していくのに100個近くのコードを処理していかなければなりません。この、処理はステータスの数だけやらないといけなく、今回、HPや攻撃力、防御力など全部で合わせて7つのステータスがあるためレベルが上がるたびに700個近くのコードを実行しなければいけません。そこで二分探索を使ってコードの数を減らしていきます。

この方法はもし、プレイヤーのレベルが50以上なら左49以下なら右へ、次に25以上なら左24以下なら右という風に処理をしていくことでコードの数を減らしていきます。



このような処理をしていくことでコードの数を減らすことができます。

### 3.3 それぞれのステータス

ここからは、それぞれのステータス、XP、HP、攻撃力、防御力、自動回復回、避率、会心率、会心ダメージについて解説していきます。下の表がステータスの数値です。

Lv	xp	HP	攻撃力	防御力	自動回復	回避率	会心率	会心ダメージ	Lv	xp	HP	攻撃力	防御力	自動回復	回避率	会心率	会心ダメージ
1	100	20	1	4	20	1%	1%	1%	51	11739	270	51	204	270	26%	26%	51%
2	110	25	2	8	25	1%	1%	2%	52	12913	275	52	208	275	26%	26%	52%
3	121	30	3	12	30	2%	2%	3%	53	14204	280	53	212	280	27%	27%	53%
4	133	35	4	16	35	2%	2%	4%	54	15625	285	54	216	285	27%	27%	54%
5	146	40	5	20	40	3%	3%	5%	55	17187	290	55	220	290	28%	28%	55%
6	161	45	6	24	45	3%	3%	6%	56	18906	295	56	224	295	28%	28%	56%
7	177	50	7	28	50	4%	4%	7%	57	20797	300	57	228	300	29%	29%	57%
8	195	55	8	32	55	4%	4%	8%	58	22876	305	58	232	305	29%	29%	58%
9	214	60	9	36	60	5%	5%	9%	59	25164	310	59	236	310	30%	30%	59%
10	236	65	10	40	65	5%	5%	10%	60	27680	315	60	240	315	30%	30%	60%
11	259	70	11	44	70	6%	6%	11%	61	30448	320	61	244	320	31%	31%	61%
12	285	75	12	48	75	6%	6%	12%	62	33493	325	62	248	325	31%	31%	62%
13	314	80	13	52	80	7%	7%	13%	63	36842	330	63	252	330	32%	32%	63%
14	345	85	14	56	85	7%	7%	14%	64	40527	335	64	256	335	32%	32%	64%
15	380	90	15	60	90	8%	8%	15%	65	44579	340	65	260	340	33%	33%	65%
16	418	95	16	64	95	8%	8%	16%	66	49037	345	66	264	345	33%	33%	66%
17	459	100	17	68	100	9%	9%	17%	67	53941	350	67	268	350	34%	34%	67%
18	505	105	18	72	105	9%	9%	18%	68	59335	355	68	272	355	34%	34%	68%
19	556	110	19	76	110	10%	10%	19%	69	65268	360	69	276	360	35%	35%	69%
20	612	115	20	80	115	10%	10%	20%	70	71795	365	70	280	365	35%	35%	70%
21	673	120	21	84	120	11%	11%	21%	71	78975	370	71	284	370	36%	36%	71%
22	740	125	22	88	125	11%	11%	22%	72	86872	375	72	288	375	36%	36%	72%
23	814	130	23	92	130	12%	12%	23%	73	95559	380	73	292	380	37%	37%	73%
24	895	135	24	96	135	12%	12%	24%	74	105115	385	74	296	385	37%	37%	74%
25	985	140	25	100	140	13%	13%	25%	75	115627	390	75	300	390	38%	38%	75%
26	1083	145	26	104	145	13%	13%	26%	76	127190	395	76	304	395	38%	38%	76%
27	1192	150	27	108	150	14%	14%	27%	77	139908	400	77	308	400	39%	39%	77%
28	1311	155	28	112	155	14%	14%	28%	78	153899	405	78	312	405	39%	39%	78%
29	1442	160	29	116	160	15%	15%	29%	79	169289	410	79	316	410	40%	40%	79%
30	1586	165	30	120	165	15%	15%	30%	80	186218	415	80	320	415	40%	40%	80%
31	1745	170	31	124	170	16%	16%	31%	81	204840	420	81	324	420	41%	41%	81%
32	1919	175	32	128	175	16%	16%	32%	82	225324	425	82	328	425	41%	41%	82%
33	2111	180	33	132	180	17%	17%	33%	83	247856	430	83	332	430	42%	42%	83%
34	2323	185	34	136	185	17%	17%	34%	84	272642	435	84	336	435	42%	42%	84%
35	2555	190	35	140	190	18%	18%	35%	85	299906	440	85	340	440	43%	43%	85%
36	2810	195	36	144	195	18%	18%	36%	86	329897	445	86	344	445	43%	43%	86%
37	3091	200	37	148	200	19%	19%	37%	87	362887	450	87	348	450	44%	44%	87%
38	3400	205	38	152	205	19%	19%	38%	88	399175	455	88	352	455	44%	44%	88%
39	3740	210	39	156	210	20%	20%	39%	89	439093	460	89	356	460	45%	45%	89%
40	4114	215	40	160	215	20%	20%	40%	90	483002	465	90	360	465	45%	45%	90%
41	4526	220	41	164	220	21%	21%	41%	91	531302	470	91	364	470	46%	46%	91%
42	4979	225	42	168	225	21%	21%	42%	92	584432	475	92	368	475	46%	46%	92%
43	5476	230	43	172	230	22%	22%	43%	93	642876	480	93	372	480	47%	47%	93%
44	6024	235	44	176	235	22%	22%	44%	94	707163	485	94	376	485	47%	47%	94%
45	6626	240	45	180	240	23%	23%	45%	95	777880	490	95	380	490	48%	48%	95%
46	7289	245	46	184	245	23%	23%	46%	96	855668	495	96	384	495	48%	48%	96%
47	8018	250	47	188	250	24%	24%	47%	97	941234	500	97	388	500	49%	49%	97%
48	8820	255	48	192	255	24%	24%	48%	98	1035358	505	98	392	505	49%	49%	98%
49	9702	260	49	196	260	25%	25%	49%	99	1138894	510	99	396	510	50%	50%	99%
50	10672	265	50	200	265	25%	25%	50%	100	1252783	515	100	400	515	50%	50%	100%

## ・XP

まず、XP 経験値について解説していきます。経験値は2種類あり、プレイヤーが持っている必要な経験値量とモンスターが持っている経験値があります。モンスターが持っている経験値はモンスターを倒したときに手に入るもので、モンスターのレベルに応じて経験値の量が上がっていきます。そして、経験値量はレベルに比例して上がっていき以下のような式になっています。

プレイヤーの経験値量

$$\sum_{k=100}^n (k \times 1.1)$$

モンスターの経験値量

$$\sum_{k=25}^n (k \times 1.1)$$

このように自分と同じレベルのモンスターを大体3から4体倒すと1レベル上がるぐらいに調節してあります。

そして、この経験値の量ですがこの後、回避率のところの説明する乱数を用いて30%の確率で0.9倍され40%の確率で等倍、そして30%の確率で1.1倍されます。これは、毎回同じ量の経験値だと代わり映えがしないのでランダムで増減するようにしました。

## ・HP

次に、HP について解説していきます。HP もプレイヤーの HP とモンスターの HP の 2 種類あります。

まずはプレイヤーの HP について説明します。HP ですがこれは、Minecraft 内に元からある HP を使ってもよかったのですが、HP の残り残量を数字での表示、自動回復の仕組みを追加できないので今回は、HP の引数を作りそれに既存の HP の最大値を変更させて HP バーの代わりに使うことにしました。HP の計算の仕方は現在の HP から受けたダメージ量を引いていき HP が 0 になったら死亡させるという処理になっています。ただこの処理の仕方だと既存の HP 以上のダメージを受けてしまうと新たに追加した方の HP が残っていても死亡してしまいます。そこで、既存の HP + 新たに追加した HP にしています。こうすることである程度のダメージを受けても死にくくなりました。

次にモンスターの HP について説明していきます。モンスターの HP ゲージですが、これは下の URL の方の動画を参考にして、HP バーを追加しました。ただ下の方の HP バーを追加下しただけでなくいろいろな要素を追加しました。まず、HP バーのテクスチャを変更しました。ほかにもこの後出てくる会心率や会心ダメージを適応させるためのコードを追加、レベルに応じて HP の最大値が変わるようにもしました。

## ・攻撃力

次に攻撃力について解説していきます。攻撃力もほかのステータスと同じようにプレイヤー側とモンスター側の 2 種類あります。

まずは、プレイヤー側の攻撃力の説明をしていきます。これは、レベルに応じたプレイヤーの基礎攻撃力を変更させるというものです。これは、ただこの後の会心率や、会心ダメージのために基礎値を変更するだけでなく引数の方の攻撃力も作っておく必要があります。

次に、モンスター側の攻撃力こちらもプレイヤー側と同じようにレベルに応じたモンスターの基礎攻撃力を変えるというものです。最初はモンスター側の攻撃力は引数で処理をしようと考えていました。しかし、攻撃してきたモンスターを指定する処理の仕方が難しく断念してしまいました。引数で攻撃力を変更することができていればプレイヤー側の HP の最大値を多めに設定しなくてよくなり見栄えもよくなったので残念でした。

## ・防御力

次に防御力について解説していきます。防御力もほかのステータスと同じようにプレイヤー側とモンスター側の2種類あります。

防御力は、受けたダメージを割合で軽減するというものです。ただ、Minecraft内のコードでは、引数に小数点以下の数字を入れることができず、あらかじめ100倍した数値を入れて処理をするときに100で割り小数点以下の数値も計算するという方法を使いました。例えば、受けたダメージが100でダメージを45%軽減したいには下のような計算をしていきます。

普通の場合

$$100 \times 0.65 = 65$$

Minecraftの場合

$$0.65 \times 100 = 65$$

$$100 \times 65 = 6500$$

$$6500 \div 100 = 65$$

もし、小数点以下が出た場合は切り捨てされた値が入ります。

次にモンスターの場ですがこれは、プレイヤー側と全く同じ処理をしています。

## ・自動回復

次に自動回復について解説していきます。自動回復はプレイヤーだけのステータスです。

自動回復の機能を作るにはまず、タイマーを作る必要があります。タイマーの作り方が、Minecraftでは、tickというものがありこれは、1ティックごとに入力したコードを実行し続けるというものです。この実行するスピードですが、大体1秒に20回ほど実行します。この機能を利用しカウンタという引数を作り20回1を足してもカウンタが20になったらカウンタから20を引きあるコードを実行させるという形にすれば1秒に1回コードを実行させることができます。

そして、このタイマーを使って自動回復の仕組みを作っていきます。1ティックごとに足していく数をレベルに応じて増やしていきます。そして、もしカウンタが2000になったらカウンタから2000を引いてある現在のHPがMAXでなければ1回復させるというコードを書きます。こうすることでレベルが上がるごとに回復させるスピードが上がっていくという仕組みになります。



## ・回避率

次に回避率について説明していきます。このステータスも自動回復と同じでプレイヤーにしかないステータスです。

まず、回避率ですがこれはプレイヤーが攻撃を受けた際確率で攻撃を回避するというものです。回避率を作るには、まず乱数を生成する必要があり乱数の作り方については下の URL の方の動画を参考にしました。乱数の仕組みですが、まず Minecraft 内にはエンティティというものがあります。エンティティにはそれぞれ UUID というランダムな ID を持っています。そこで、その ID を 100 で割りあまりの数値を引数に入れることで 0 から 99 の数値を入手することができます。このように入手した引数が 50 以上ならコードを実行し、49 以下なら実行しないという形にすることで 50% の確率でコードを実行させることが出切るようになります。

そして、この乱数を用いて回避率を作っていきます。まず、攻撃を受けた際に乱数を用いて回避するかを判別していきます。判別の仕方は、乱数がレベルごとに上がっていく数値より低いかどうかで識別し、低かった場合はダメージを 0 にする処理をします。こうすることで、レベルが上がるごとに回避率が上がっていくようになります。

## ・会心率

次に会心率について解説していきます。このステータスはプレイヤーにしかないステータスです。

まず、会心率とは、与えたダメージが確率で上がるというものです。会心率の仕組みですが、まず攻撃を与えた際にダメージが上がるかの判別をします。判別の仕方ですが先ほど回避率と同じように乱数がレベルごとに上がっていく数値より低いかどうかで識別していき、低かった場合はこの後会心ダメージで解説するダメージを増やす処理をします。このようにしてレベルに応じて会心率が上がっていくようになります。

## ・会心ダメージ

最後に会心ダメージについて説明していきます。このステータスはプレイヤーにしかないステータスです。

まず、会心ダメージとは先ほど解説した会心率が成功際にダメージ量を割合で上昇させるというものです。次に会心ダメージの仕組みですが、会心率の処理が成功した後にダメージにレベルごとに増えていく引数をかけてダメージ量を上昇するという仕組みになっています。

### 3.4 参考資料・使用ソフト

ここからはモンスターを追加する際に使用したソフトや資料をまとめます。

・参考動画

レベルアップ処理: [https://www.youtube.com/watch?v=SuMkQ-\\_Par0&t=2s](https://www.youtube.com/watch?v=SuMkQ-_Par0&t=2s)

ステータス処理:

<https://www.youtube.com/watch?v=LcfHAcQDBjk&list=PL6gYTGrV1UW7DUQkgPzOgSW20uEqUqrLU&index=2>

モンスターの HP 表示処理:

<https://www.youtube.com/watch?v=ubWh5NU7Zg&list=PL6gYTGrV1UW7kgjBP3bqaJ0A88ZFCsu6n&index=4>

乱数生成: <https://www.youtube.com/watch?v=AjDvvOWVGOA>

・使用ソフト

Visual Studio Code: [Visual Studio Code - Code Editing. Redefined](#)

## 4.RPG マップ(ダンジョンや建築物)

### 4.1 マップ

まず、マップはワールドペインターというソフトを使って作成した。このソフトでは絵を描く感覚でRPGのマップを作ることができ6地帯ほど追加しました。



- ・草原地帯 (草原が広がり、木々がほとんどない)
- ・森林、ジャングル地帯 (たくさんの木が広がっているジャングルのような地帯)
- ・砂漠地帯 (砂が広がっており、植物がほとんど生息しない地帯)
- ・降雪地帯 (一面に雪が積もり、周りの海は凍っている)
- ・火山地帯 (ひときわ目立つおおきな火山があり、まわりには珍しい植物がある)
- ・山岳地帯 (高く険しい岩が露出した山々がそびえる地帯。)

## 4.2 ダンジョン

これまでに作ったステータスやモンスター、アイテムが活躍できるようにレベルアップができる仕組み、モンスターが出現する仕組み、宝箱の中身がランダムで変わるしくみを作りました

- ・ステータスの表示

上記で作ったステータスをレベルが上がると表示されるようにした。

- ・モンスターが出現

プレイヤーが半径10マス以内に近づいた際にモンスターが召喚される仕組みを作成した。

- ・宝箱

ダンジョンに入りなおすたびに宝箱の中身がランダムに変わる仕組みを作りました。

## 4.3 建築物

マップの各所にその地域に適した建築物をメンバーのそれぞれが建築しました。

# 成果・結果

今回、活動の目的である RPG ゲームの仕組みを理解することと、チームでゲームを開発するという大変を学ぶことができた。まず、RPG ゲームの仕組みについては上記で説明したステータスやダンジョンを追加する際に理解することができた。しかし、今回追加したかったが時間が足りず追加できなかつた要素がある。それは、魔法やスキル、ショップやドロップアイテム、さらにボスモンスター要素だ。これらの要素は追加することはできなかつたがどのようにすれば追加できるかは理解することはできた。

次に、チームでの開発することについては、メリットとデメリットがあることを学ぶことができた。まずメリットは、長期的に見ればひとりで開発するよりもチームで開発することができるが増えるということだ。しかし、今回は活動する期間が短かつたためこのメリットを実感することは少なかつた。次にデメリットは、情報を伝達することが難かつたということだ。今回は、一度に情報を共有するために口頭ではなく Word を使って資料を作り理解できなかつたところを何度も見返せるようにした。