



minecraft
GLOWSTONE

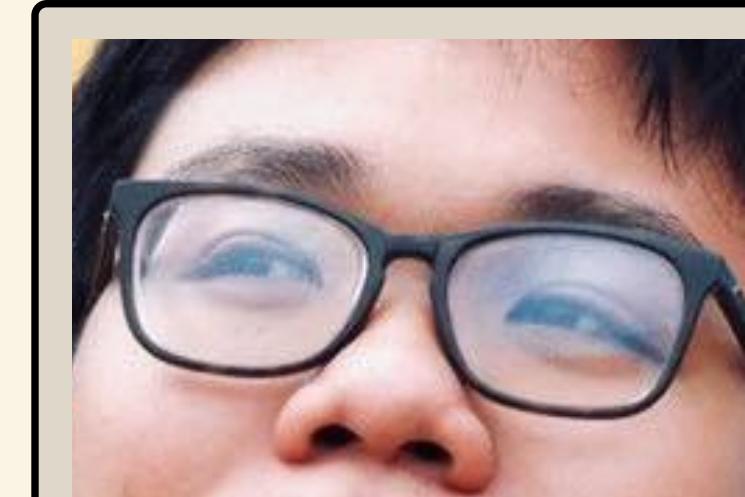
BY GROUP 4

MEMBERS

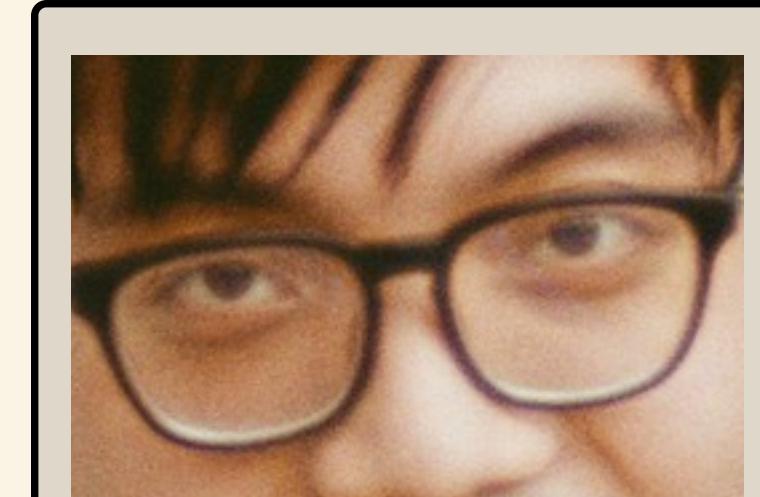
Page 02 of 30



61010345 NUTTAPON



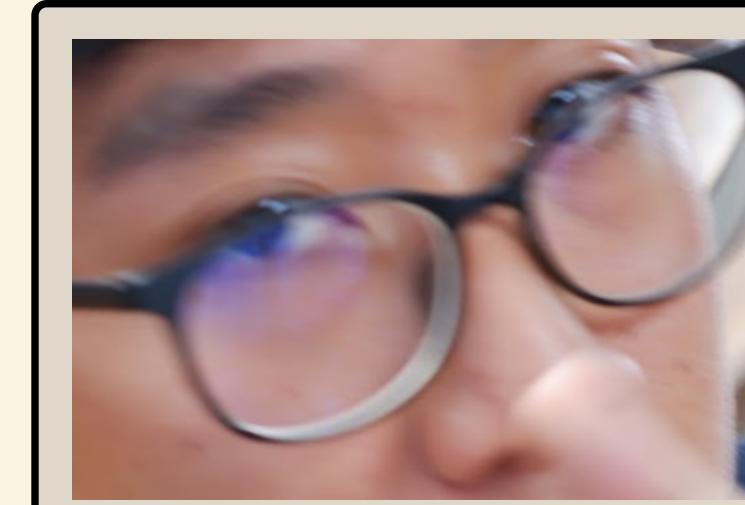
61010362 NATASIT



61010402 TANTATORN



61010707 PASAWEE



61010750 PIPITPONG



61010827 PHUMPHATHAI

~~Table of~~ **CONTENT**



WHAT IS GLOWSTONE ?

ARCHITECTURE

QUALITY ATTRIBUTES

DESIGN PATTERN

WHAT IS GLOWSTONE ?

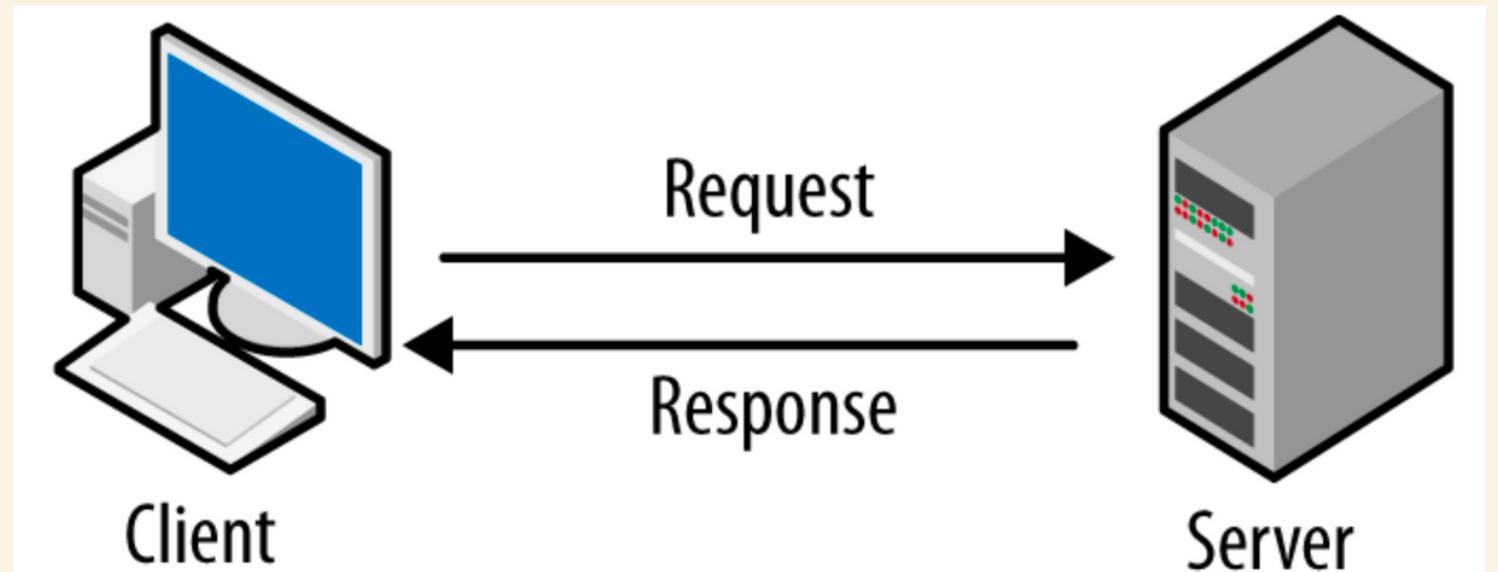
Glowstone เป็น Lightweight Open-Source Minecraft Server ที่พัฒนาโดยใช้ภาษา Java เขียนขึ้นมาใหม่ทั้งหมด โดยไม่นำเอาโค้ดภายใน Minecraft มาใช้เป็นรากฐาน (Decompile) ทำให้มี functionality ที่ไม่จำเป็นจากต้นฉบับ มีประสิทธิภาพที่สูงขึ้นได้ ง่ายต่อการปรับปรุงแก้ไข และยังคงความสามารถในการใช้งาน Plugin จาก Bukkit API, Spigot, Paper, และ fork ต่าง ๆ หมายเหตุรับเซิร์ฟเวอร์ที่ต้องการการรองรับผู้เล่นจำนวนมาก แต่ไม่ต้องการการปรับแต่งเยอะ



ARCHITECTURE

CLIENT-SERVER PATTERN

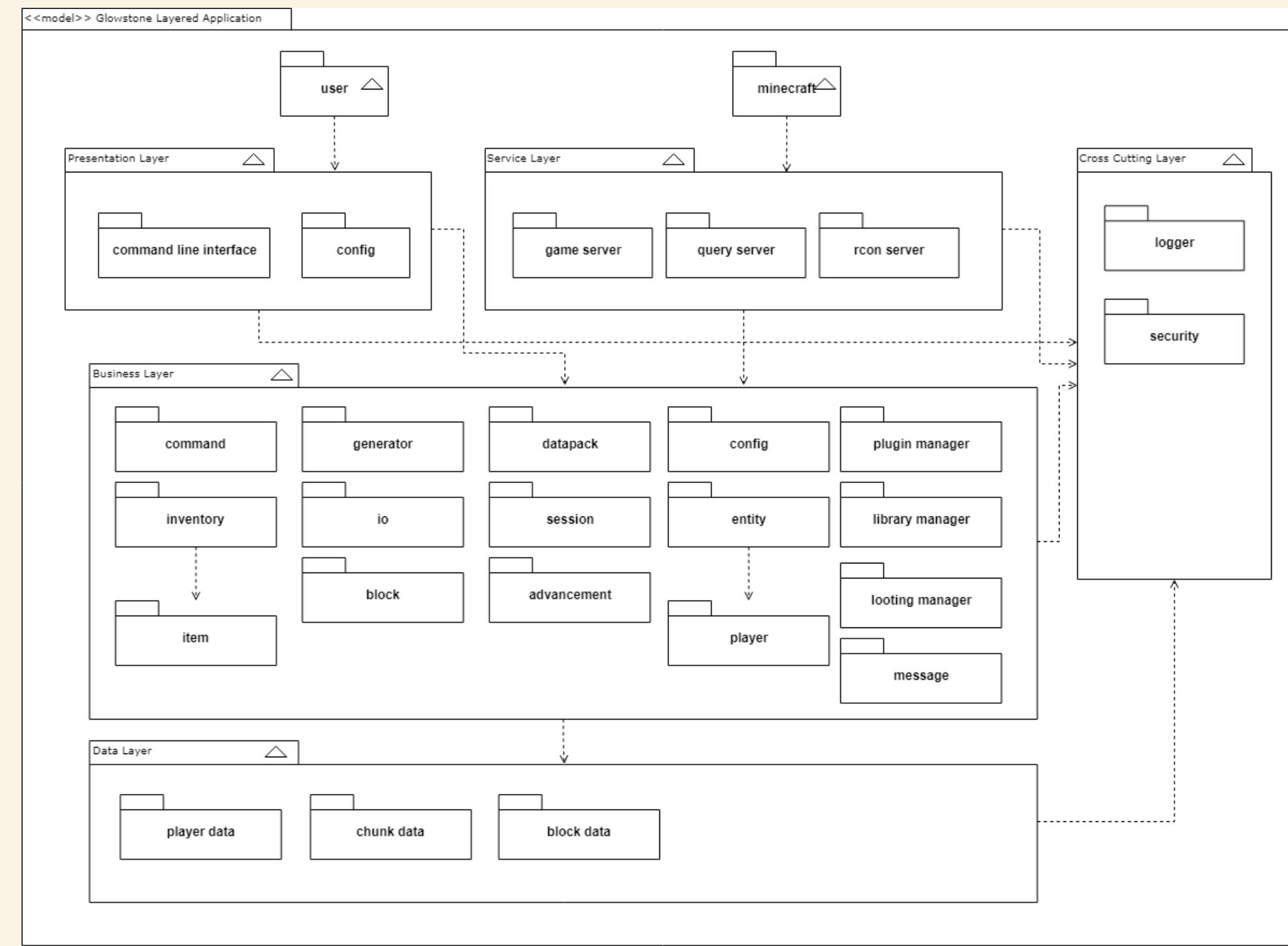
Glowstone เป็น Minecraft Server สำหรับให้ Minecraft Client มาเชื่อมต่อ เพื่อเข้าเล่น



MICROKERNEL PATTERN



LAYERED PATTERN



WEAKNESS

SINGLE POINT OF FAILURE

Glowstone เก็บข้อมูลแบบ **Stateful**
(data ทั้งหมดถูกเก็บรวมไว้ในที่เดียว) ทั้งยังมี **instance** เดียว

วิธีแก้

เปลี่ยนวิธีการเก็บข้อมูลไปใช้ **Database** แทน
และทำ **Data Replication** ไปที่เครื่องสำรองในเครือข่าย

สามารถเพิ่ม server instance และ proxy server

สำหรับสลับ server ให้ client อัตโนมัติ

เพื่อให้มี **Availability** และ **Failure Transparency**

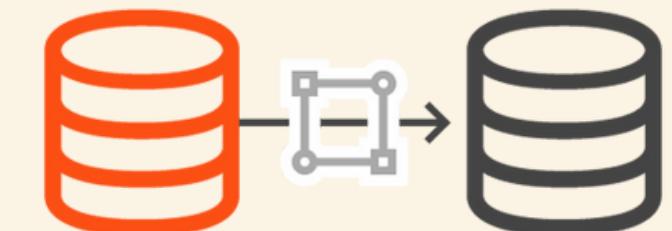
&

NOT ENOUGH SCALABILITY

Glowstone เป็น **Stateful**
ไม่สามารถ scale ตัว server ได้
กรณีมี inbound request จำนวนมาก

วิธีแก้

เปลี่ยน Glowstone ให้เป็น Stateless
โดยการเก็บข้อมูลใน Database แทน



quality
ATTRIBUTE

CONFIGURABILITY

Glowstone รองรับการตั้งค่าแบบด้วยไฟล์รูปแบบต่างๆ เช่น
JSON, XML หรือ Text

- Tactics : Glowstone มีการ detect ไฟล์ตั้งค่าเพื่อ
นำมายังไฟล์ configuration ไม่ต้อง compile code ใหม่

```
public ServerConfig(File directory, File configFile, Map<Key, Object> parameters) {
    checkNotNull(directory);
    checkNotNull(configFile);
    checkNotNull(parameters);

    this.directory = directory;
    this.configFile = configFile;
    this.parameters = parameters;

    config.options().indent(4).copyHeader(true).header(
        "glowstone.yml is the main configuration file for a Glowstone server\n"
        + "It contains everything from server.properties and bukkit.yml in a\n"
        + "normal CraftBukkit installation.\n\n"
        + "Configuration entries are documented on the wiki: "
        + "https://docs.glowstone.net/en/latest/Configuration_Guide/index.html\n"
        + "For help, join us on Discord: https://discord.gg/TFJqhsC");
}

///////////////////////////////
// Modification

/**
 * Save the configuration back to file.
 */
public void save() {
    try {
        config.save(configFile);
    } catch (IOException e) {
        GlowServer.logger.log(Level.SEVERE, "Failed to write config: " + configFile, e);
    }
}
```

COMPATIBILITY

Glowstone รองรับการใช้งานร่วมกับ Bukkit, Spigot และ Paper plugins เพียงแค่นำไฟล์ .jar ไว้ใน plugins directory

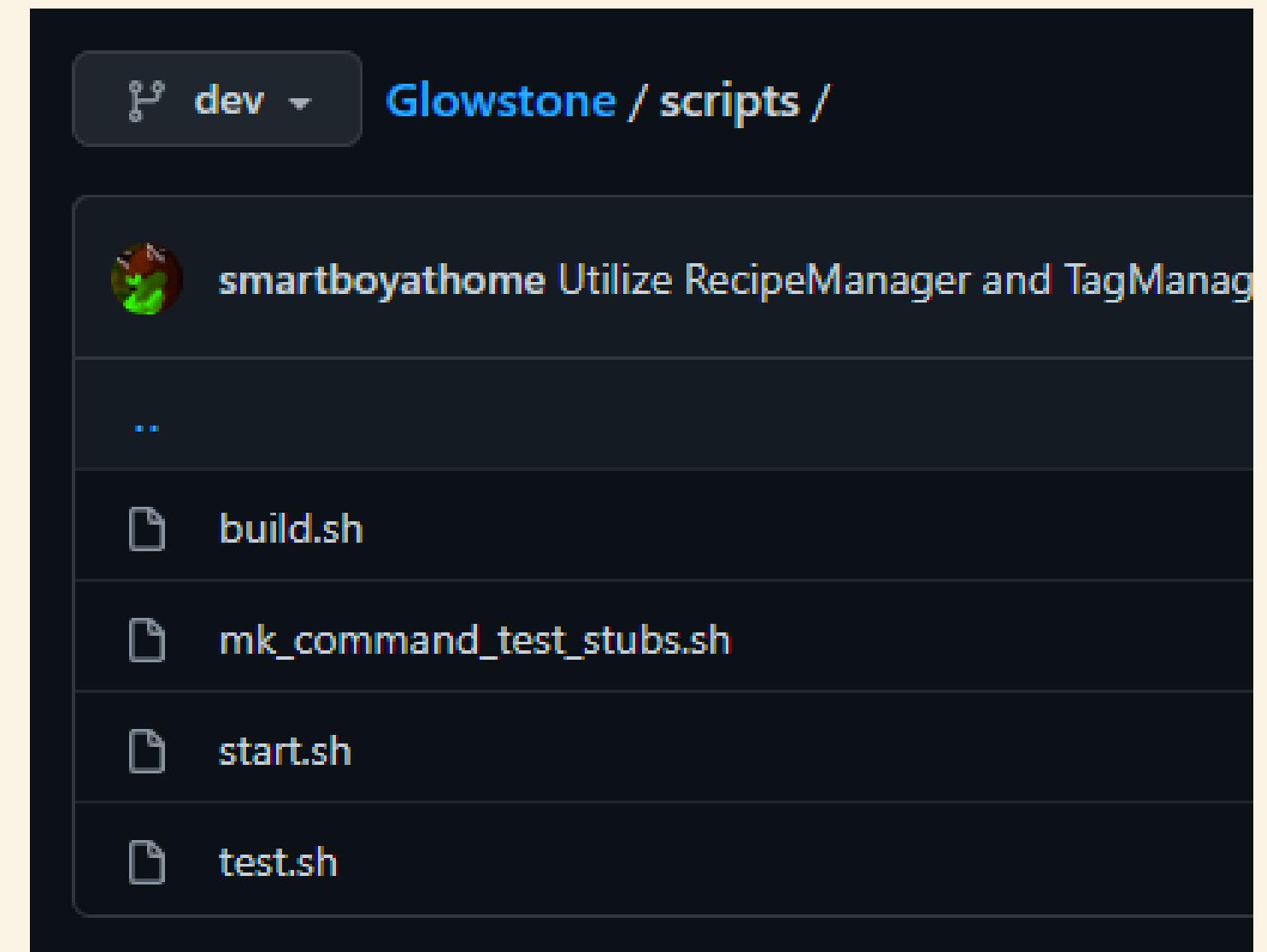
Tactics : Glowstone มีการ detect ว่าไฟล์ .jar มาจาก plugin ตัวใด เพื่อนำไปใช้ในเกม Minecraft ร่วมกับตัว Glowstone server

```
1285     commandMap.register( fallbackPrefix: "minecraft", new ToggleDownfallCommand());
1286     commandMap.register( fallbackPrefix: "minecraft", new SetWorldSpawnCommand());
1287     commandMap.register( fallbackPrefix: "minecraft", new PlaySoundCommand());
1288     commandMap.register( fallbackPrefix: "minecraft", new EffectCommand());
1289     commandMap.register( fallbackPrefix: "minecraft", new EnchantCommand());
1290     commandMap.register( fallbackPrefix: "minecraft", new TestForCommand());
1291     commandMap.register( fallbackPrefix: "minecraft", new TestForBlockCommand());
1292     commandMap.register( fallbackPrefix: "minecraft", new SetBlockCommand());
1293     commandMap.register( fallbackPrefix: "minecraft", new CloneCommand());
1294     commandMap.register( fallbackPrefix: "minecraft", new TestForBlocksCommand());
1295
1296     File folder = new File(config.getString(Key.PLUGIN_FOLDER));
1297     if (!folder.isDirectory() && !folder.mkdirs()) {
1298         ConsoleMessages.Error.Plugin.MKDIR.log(folder);
1299     }
1300
1301     // detect plugin types
1302     pluginTypeDetector = new GlowPluginTypeDetector(folder);
1303     pluginTypeDetector.scan();
1304
1305     // scan plugins for @Field and @Box annotated fields
1306     FieldSet annotatedFields = new FieldSet();
1307     Boxes boxes = new Boxes();
1308     LinkstoneRuntimeData.setFields(annotatedFields);
1309     LinkstoneRuntimeData.setBoxes(boxes);
1310     new LinkstonePluginScanner(annotatedFields, boxes)
1311         .scanPlugins(pluginTypeDetector.bukkitPlugins);
```

USABILITY

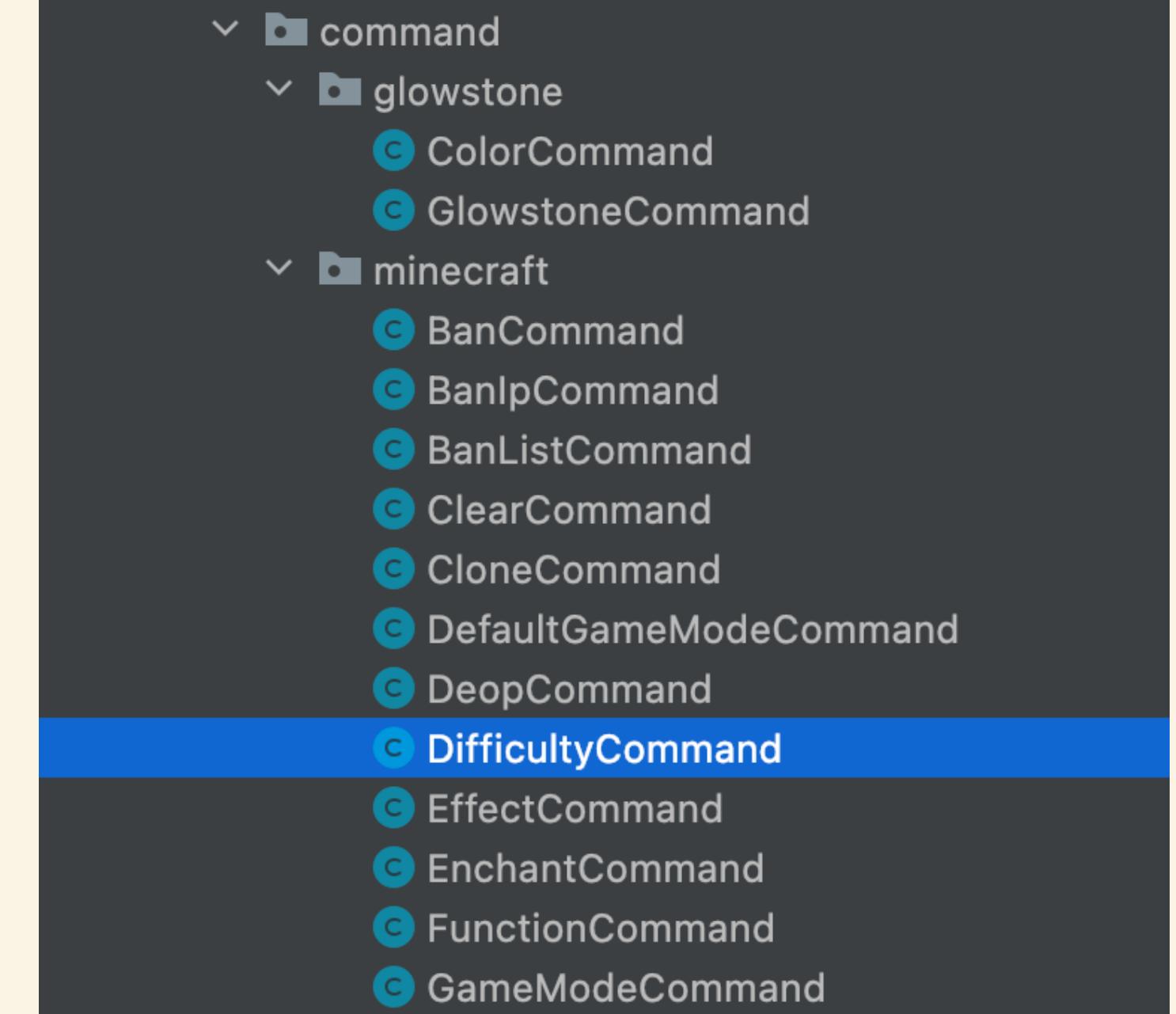
Glowstone มี CLI ในการใช้สั่งรัน server จากทาง Terminal รวมทั้งการกระทำอื่นๆ เช่น เตะคนออกจาก server

Tactics : Glowstone มีการ print สถานะการทำงานบน Terminal มี script สำหรับทั้งการ build และ start server



OPERABILITY

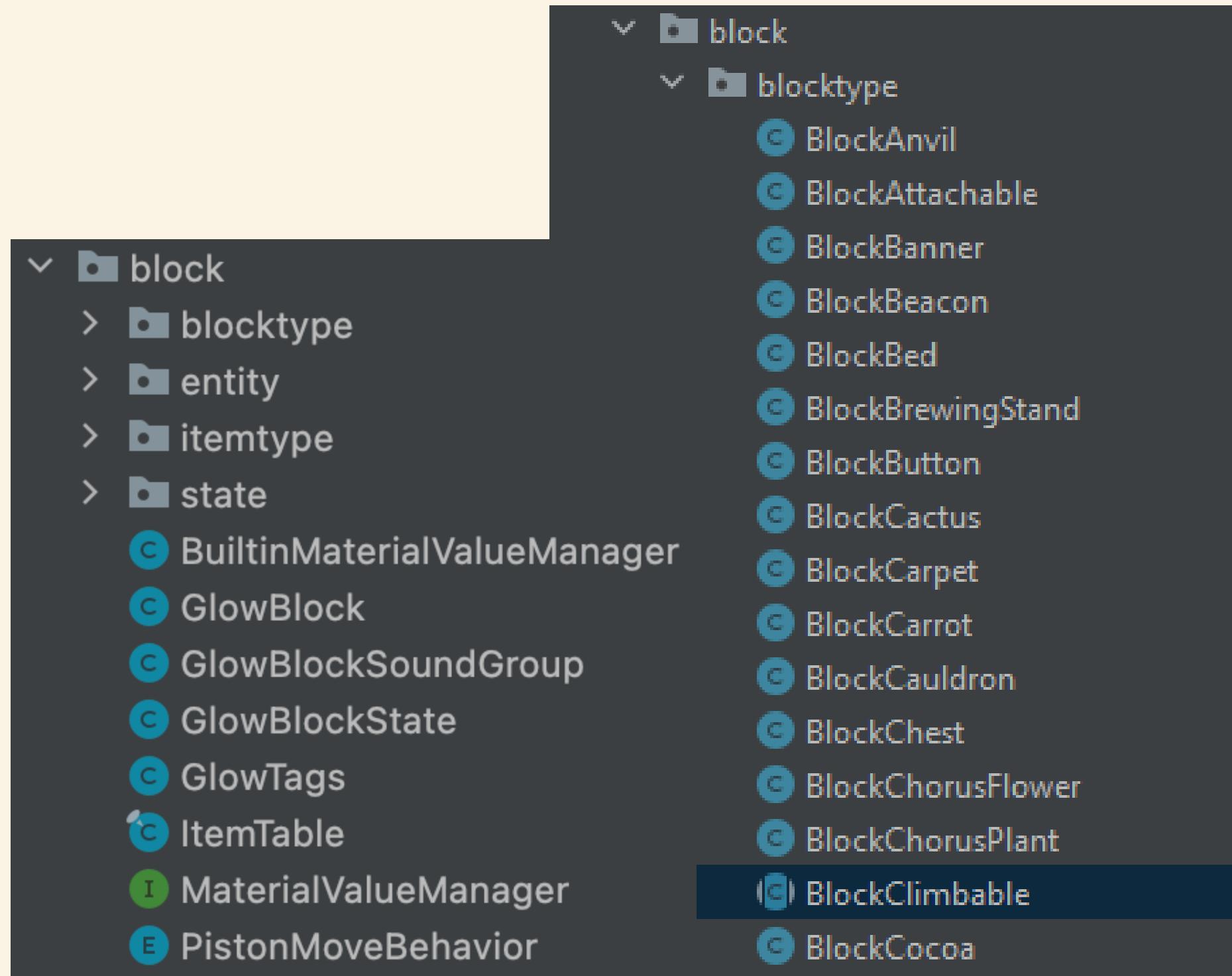
Tactics : Glowstone มีการสร้างคำสั่งสำหรับให้ admin
ควบคุมเซิร์ฟเวอร์สามารถใช้งานผ่านทาง command
ภายในตัวเกม



MODIFIABILITY

Glowstone มีการเขียน class ที่ยึดหยุ่นเหมาะสมแก้การที่นักพัฒนาสามารถเพิ่มเติมหรือแก้ไขได้ง่าย

Tactics : Glowstone มีการเขียน abstract class ของแต่ละชนิดไว้สำหรับนักพัฒนาที่ต้องการเพิ่ม Block ใหม่โดยการ extends abstract class ไปพัฒนาต่อได้ง่าย



design
PATTERN

FACTORY PATTERN

บรรทัดที่ 2422:2446 ของ class GlowServer

มี method getGenerator ที่จะสร้าง object ตามค่าที่ส่งเข้าไปโดยแต่ละ object ที่สร้างจะเป็น ChunkGenerator เมื่อันกัน

ไฟล์ที่เกี่ยวข้อง

- <https://github.com/GlowstoneMC/Glowstone/blob/dev/src/main/java/net/glowstone/GlowServer.java>

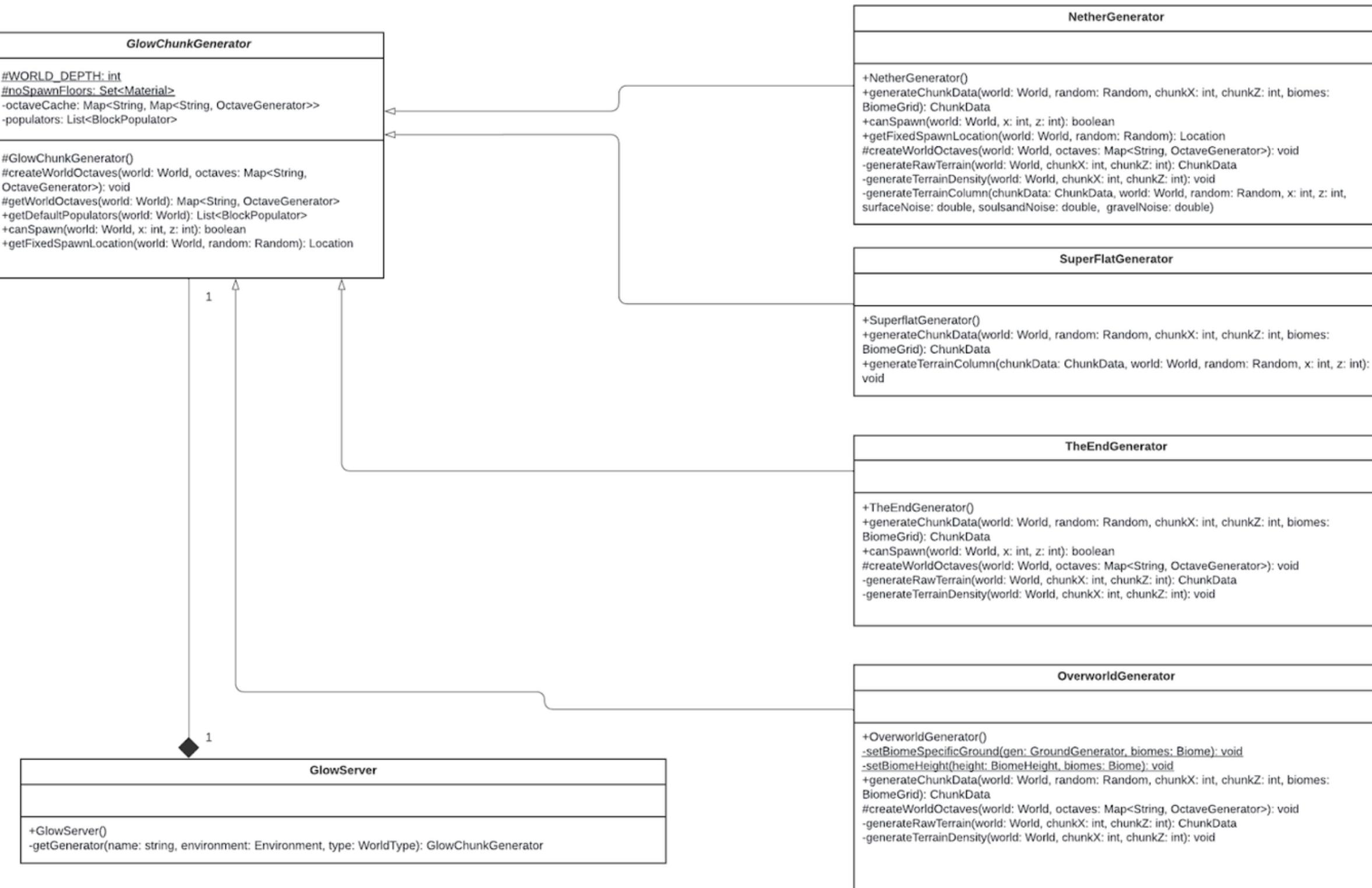
ที่มา: https://www.tutorialspoint.com/design_pattern/factory_pattern.htm

```
/*
 * 
private ChunkGenerator getGenerator(String name, Environment environment, WorldType type) {
    // find generator based on configuration
    ConfigurationSection worlds = config.getWorlds();
    if (worlds != null) {
        String genName = worlds.getString(name + ".generator", null); // NON-NLS
        ChunkGenerator generator = WorldCreator
            .getGeneratorForName(name, genName, getConsoleSender());
        if (generator != null) {
            return generator;
        }
    }

    // find generator based on environment and world type
    if (environment == Environment.NETHER) {
        return new NetherGenerator();
    } else if (environment == Environment.THE_END) {
        return new TheEndGenerator();
    } else {
        if (type == WorldType.FLAT) {
            return new SuperflatGenerator();
        } else {
            return new OverworldGenerator();
        }
    }
}
```

Glowstone: Factory Pattern

Tantatorn Suksangwan | April 20, 2022



SINGLETON PATTERN

บรรทัดที่ 176:178 ใน class ItemTable

มี method instance เมื่อเรียกใช้งานจะได้ object ของ ItemTable ที่ทำการสร้างไว้ที่บรรทัดที่ 159

ไฟล์ที่เกี่ยวข้อง

- glowstone/block/ItemTable.java

```
private static final ItemTable INSTANCE = new ItemTable();

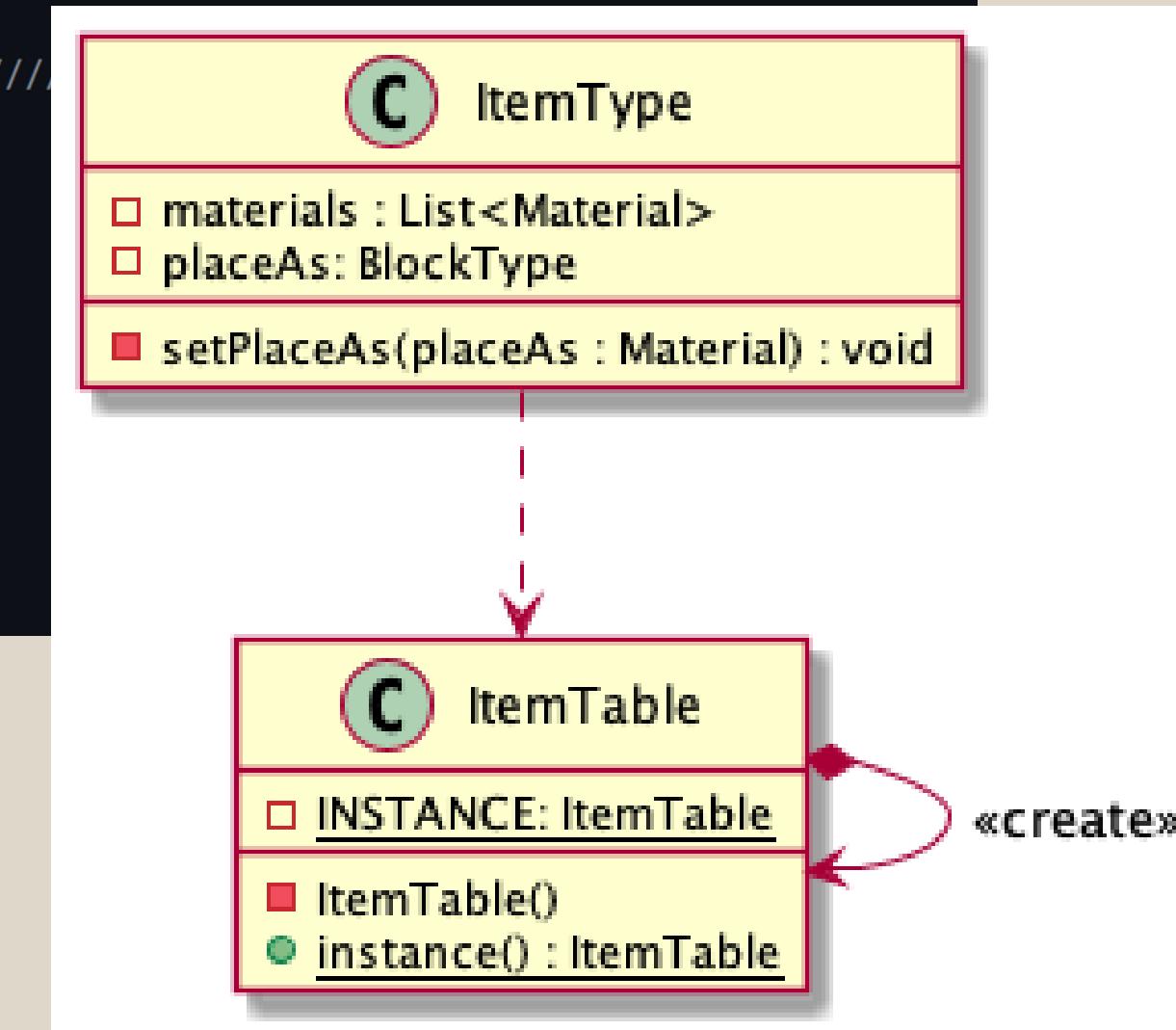
static {
    INSTANCE.registerBuiltins();
}

private final EnumMap<Material, ItemType> materialToType = new EnumMap<>(Material.class);
private final Map<NamespacedKey, ItemType> extraTypes = new HashMap<>();
private int nextBlockId;
private int nextItemId;

///////////////////////////////
// Data

private ItemTable() {
}

public static ItemTable instance() {
    return INSTANCE;
}
```



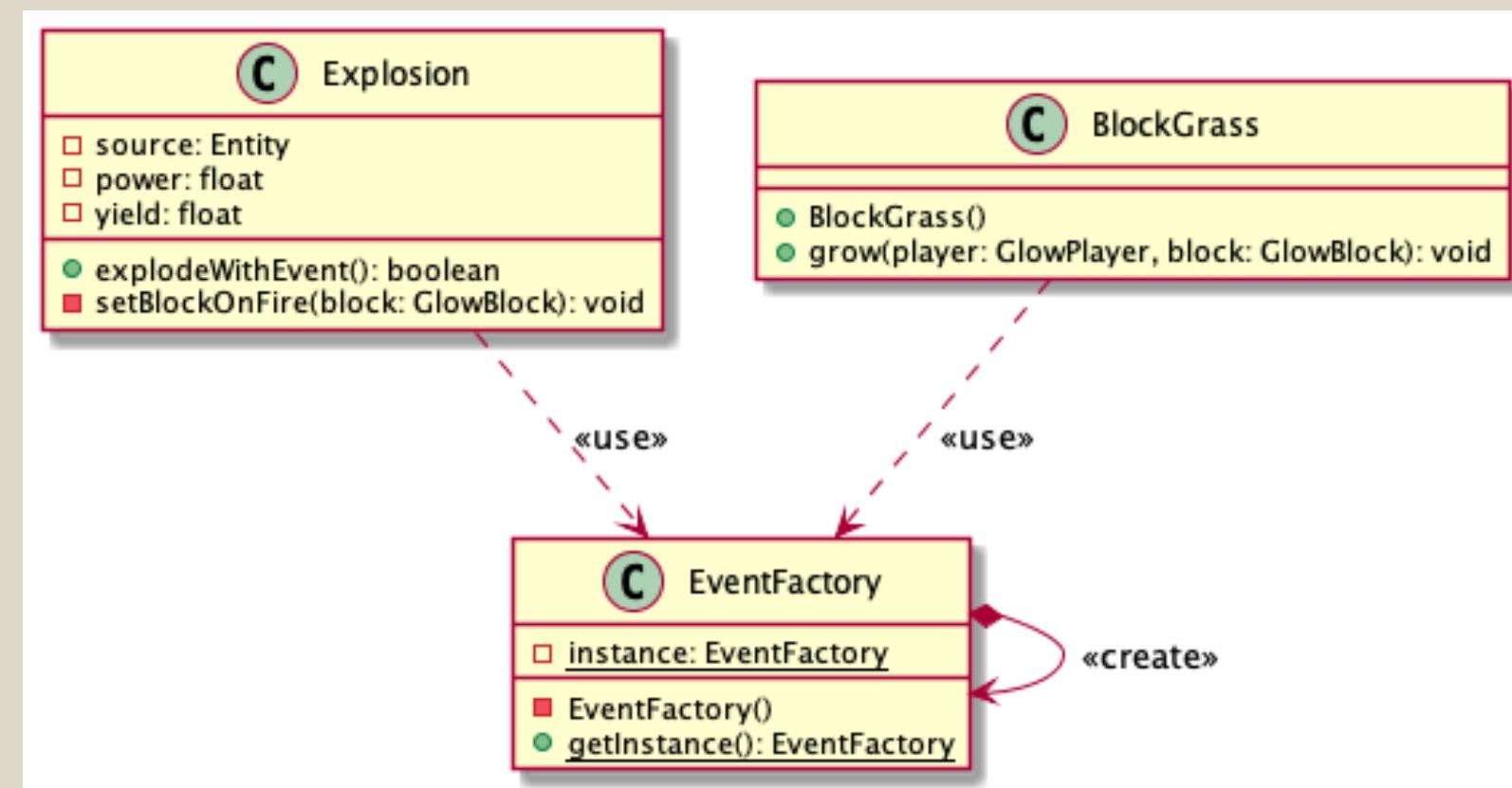
SINGLETON PATTERN

Class EventFactory ใช้ Singleton Pattern เรียกใช้ object ผ่าน EventFactory.getInstance()

ไฟล์ที่เกี่ยวข้อง

- glowstone/EventFactory.java

```
public class EventFactory {  
  
    /**  
     * The instance of this class. Setter should only be called in tests when mocking.  
     */  
    @Getter  
    @Setter  
    private static EventFactory instance = new EventFactory();  
  
    private EventFactory() {  
    }  
}
```



PROTOTYPE PATTERN

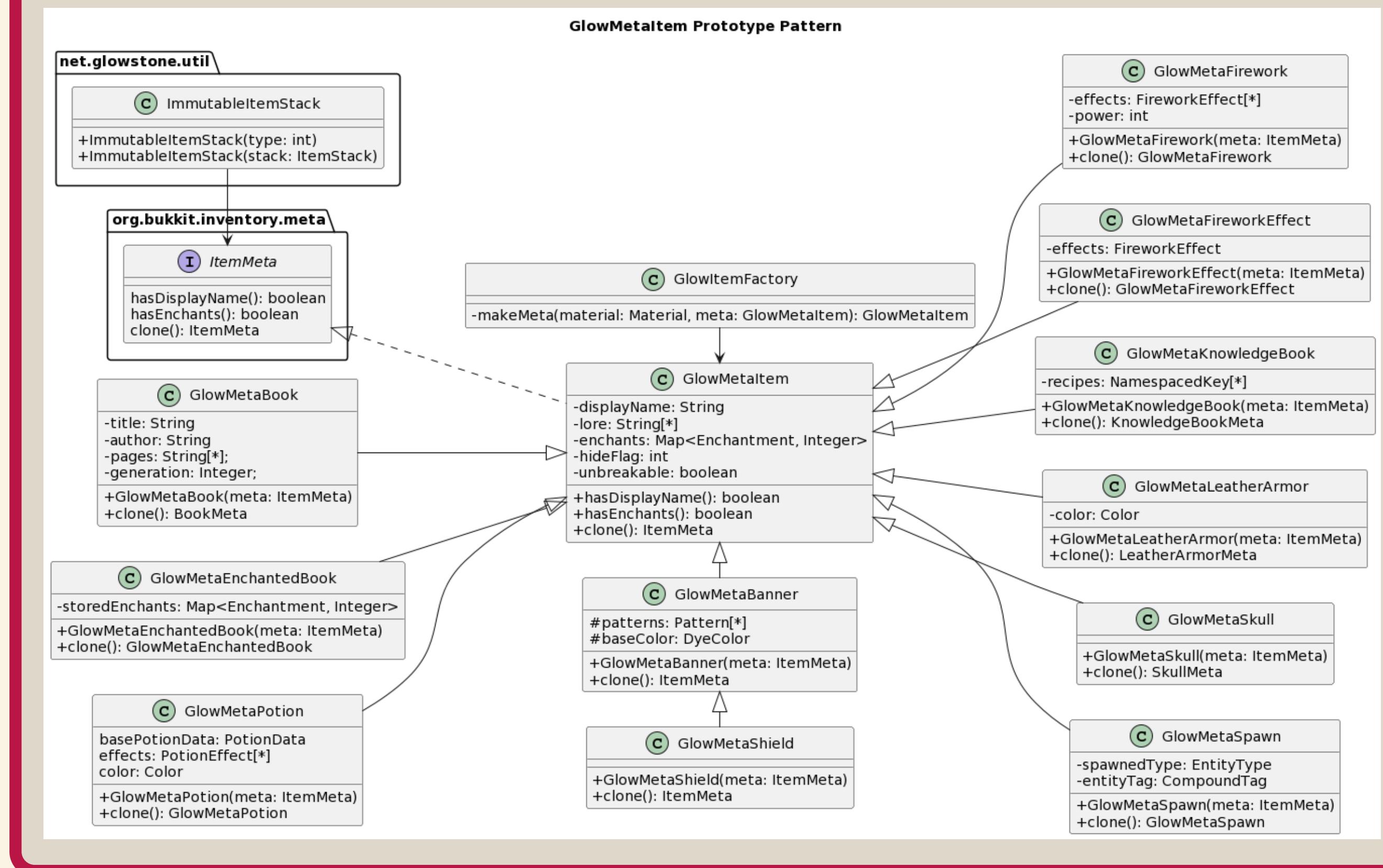
บรรทัดที่ 305 : 307 มี method clone() เพื่อคัดลอก object ออกมาเป็นอีก object หนึ่ง

ไฟล์ที่เกี่ยวข้อง

- glowstone/inventory/GlowMetalItem.java

```
303  
304     @Override  
305     public ItemMeta clone() {  
306         return new GlowMetaItem(this);  
307     }  
308
```

```
public ImmutableItemStack(ItemStack stack) throws IllegalArgumentException {  
    super(stack);  
    itemMeta = stack.getItemMeta().clone();  
}
```



OBSERVER PATTERN

class GlowScoreboard

บรรทัดที่ 65:96 มีการสร้าง method subscribe เพื่อให้ player ที่ต้องการ sub- scribe การเปลี่ยนแปลงมาเรียกใช้งาน

บรรทัดที่ 102:120 มี method unsubscribe เพื่อยกเลิกการรับการเปลี่ยนแปลงที่จะเกิดในอนาคต

บรรทัดที่ 126:131 มี method broadcast เพื่อให้ Class ที่ต้องการเปลี่ยนแปลงข้อมูลส่งข้อมูลให้ player ที่ subscribe อยู่ เช่น การเปลี่ยนแปลงคะแนน การเปลี่ยนแปลงทีมที่ผู้เล่นแต่ละคนอยู่

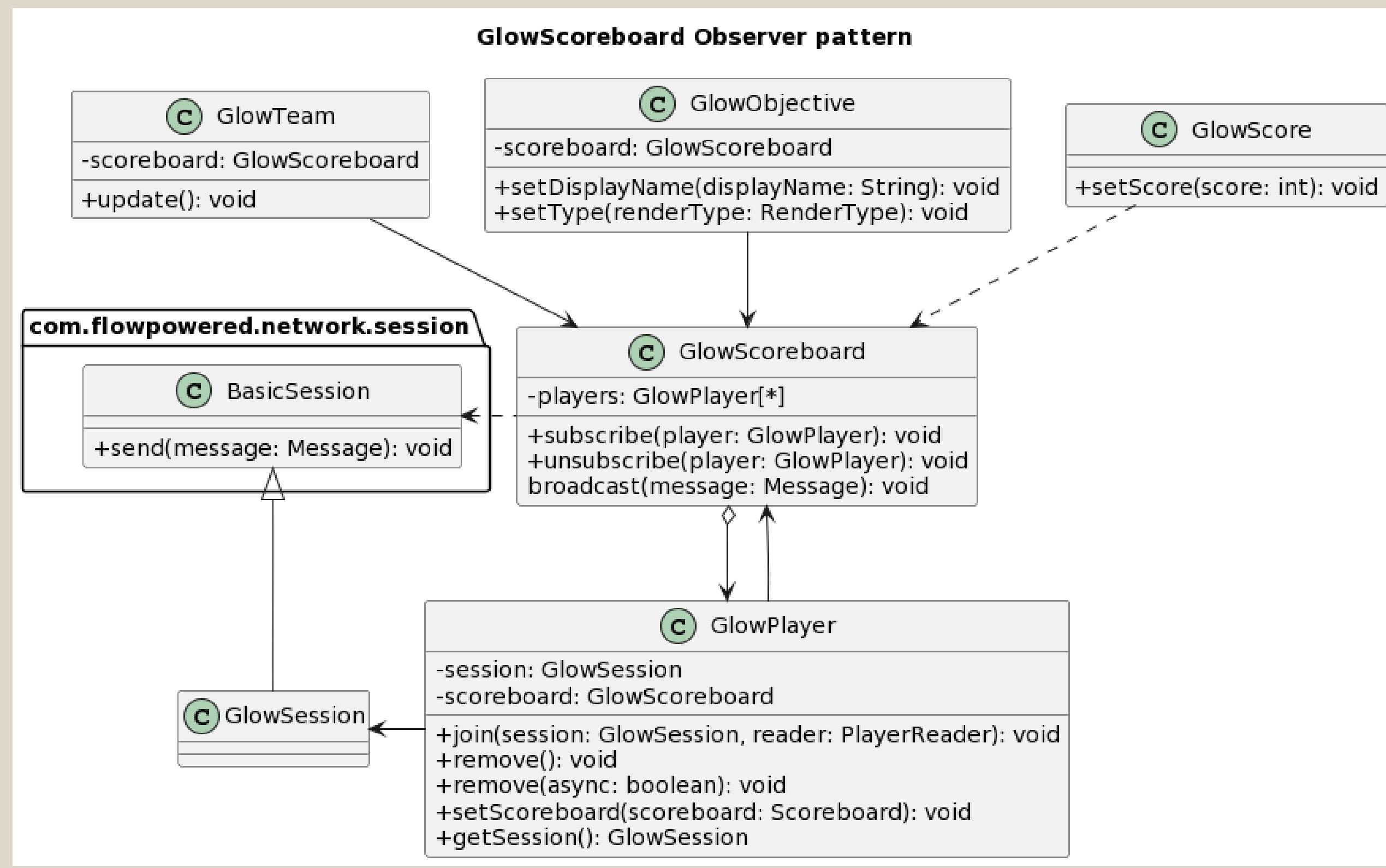
ไฟล์ที่เกี่ยวข้อง

- glowstone/scoreboard/GlowScoreboard.java

```
/**  
 * Send a player this scoreboard's contents and subscribe them to future changes.  
 *  
 * @param player The player to subscribe.  
 */  
public void subscribe(GlowPlayer player) {  
    // send all the setup stuff  
    // objectives  
    for (GlowObjective objective : objectives.values()) {  
        player.getSession().send(  
            ScoreboardObjectiveMessage.create(objective.getName(), objective.getDisplayName()));  
    }  
}
```

```
/**  
 * Clear the player's scoreboard contents and  
 *  
 * @param player The player to unsubscribe.  
 */  
public void unsubscribe(GlowPlayer player) {  
    // remove from player set  
    players.remove(player);  
}
```

```
/**  
 * Broadcast a scoreboard update to all subscribed players.  
 *  
 * @param message The message to send.  
 */  
void broadcast(Message message) {  
    for (GlowPlayer player : players) {  
        player.getSession().send(message);  
    }  
}
```



• *FaNgk yōo*
Q & A