Minecraft で学ぶ Java プログラミング
補足資料

紅花

2020 年 5 月 4 日

目次

1 はじめに

本資料は、2020 年 5 月 4 日に実施された LT 会”XYZLT”での私の発表「Minecraft で学ぶ Java プログラミング」の補足となります。これから説明する内容は、以下の文献の内容を要点を絞ったものです。

- https://www.spigotmc.org/wiki/buildtools/
- https://www.spigotmc.org/wiki/spigot-plugin-development/
- https://hub.spigotmc.org/javadocs/spigot/overview-summary.html

これらも参照しながら作業することを強くお勧めします。

2 最終目標

本資料における最終目標は、「Minecraft サーバのプラグイン」の作り方の大まかな流れを追った上で開発環境を構築することです。その過程で必要となるサーバの導入方法についても簡単に触れた上で、IntelliJ Idea を用いて Minecraft サーバのプラグインを作成する手順を説明していきます。また、環境は Windows としま

す。Mac や Linux の人は適宜読み替えてください。

3 Minecraft サーバの導入

まず、プラグインの利用が可能な Minecraft サーバ”Spigot”を導入していきましょう。導入のために、”BuildTools”というファイルが必要です。[*1]ここからダウンロードしてください。次に、ダウンロードした jar ファイル（BuildTools.jar）を実行します。jar ファイルの入っているフォルダを SHIFT キーを押しながら右クリックし、「PowerShell ウィンドウをここで開く」を選択しましょう。PowerShell を開いたら、

```
1       java -jar BuildTools.jar
```

と入力します。すると、Spigot を含むファイルのダウンロードが自動的に始まります。ダウンロードが完了した後、BuildTools.jar と同じフォルダに spigot-xxx.jar（xxx にバージョンが入る）が生成されています。これを、別の新しいフォルダにコピー&ペーストします。

次に、spigot-xxx.jar を移動したフォルダに、spigot-xxx.jar を実行するためのバッチファイルを作りましょう。メモ帳を開いて、

```
1       @echo off
2       java -Xms1024M -Xmx1024M -jar spigot-xxx.jar
3       pause
```

と入力します。保存の際、拡張子を”.bat”としてください。

作成したバッチファイルをダブルクリックで実行しましょう。すると、同じフォルダに”eula.txt”というファイルが生成されるはずです。それを開き、

```
1       eula=false
```

となっている部分を

```
1       eula=true
```

と書き換えます（Mojang の EULA に同意する）。これ以降、バッチファイルを実行することで Minecraft サーバを起動することができます。

4 Minecraft サーバのプラグインを作成する

4.1 プロジェクトの作成

Minecraft サーバのプラグインを作成していきましょう。ここでは、Maven を使ってプラグイン作成に必要なライブラリを揃えることにします。まず、IntelliJ Idea を開き、”File → New → Project...”を選択します。次に、New Project 画面で”Maven”を選択し、Next をクリックします。

[*1] どうして直接ソフトウェアをダウンロードをできないのか、と思うことだろう。このような煩雑とした作業が要求されるのには理由がある。当時のサーバサイドプログラム開発者のいざこざで DMCA テイクダウンが行われてしまったために、ソフトの直接的なダウンロードが不可能になったのだ。そこで、BuildTools.jar を元に自分の PC 上でソフトをコンパイルすることで DMCA 問題の回避をする形を取っている。

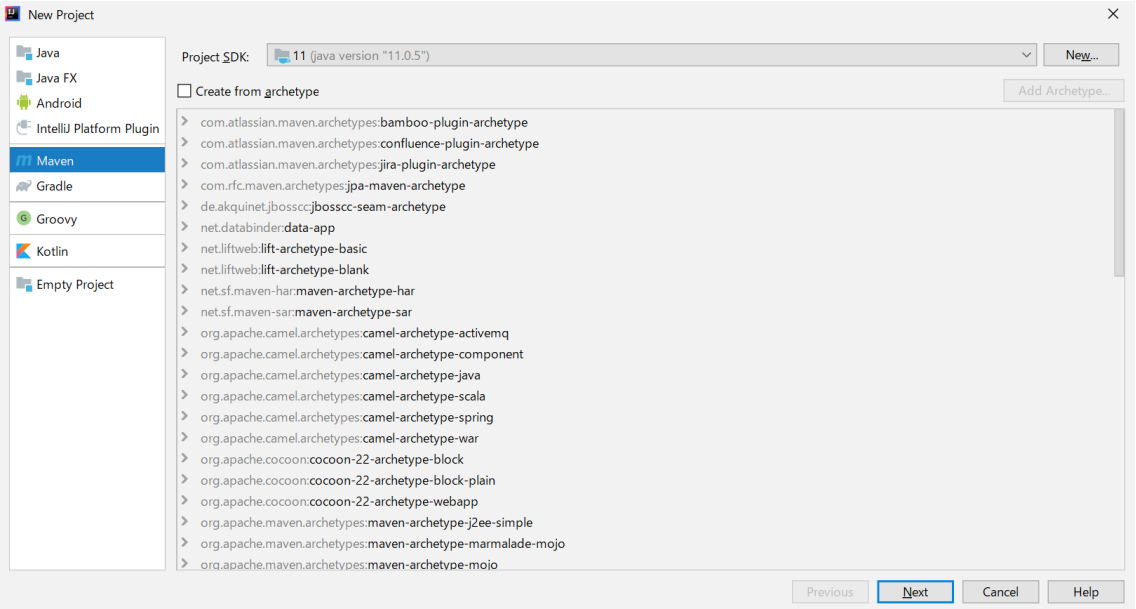

図 1

次の画面で、Name 欄には自分の作るプラグインの名前を入力します。さらに、下の”Artifact Coordinates”タブを開き、GroupId（大抵は自分のドメインを逆にしたもの:benifl.github.io ならば”io.github.benifl”のようにする）、ArtifactId（プラグインの名前）、Version（そのままでいい）を入力します。

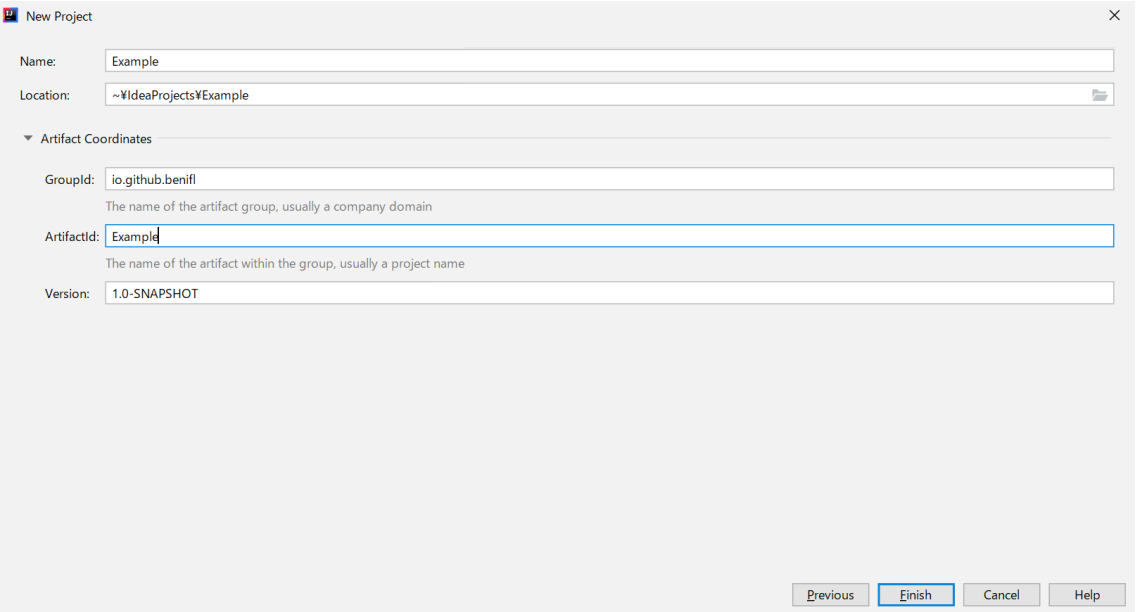

図 2

最後に Finish をクリックすればプロジェクトの作成は完了です。

次に、”pom.xml”を編集します。以下の内容を追加してください（バージョンは適宜変更してください）。

```
1           <dependencies>
2             <dependency>
3               <groupId>org.spigotmc</groupId>
4               <artifactId>spigot-api</artifactId>
5               <version>1.15.2-R0.1-SNAPSHOT</version>
6               <scope>provided</scope>
7             </dependency>
8           </dependencies>
9           <properties>
10            <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>
11            <maven.compiler.source>1.8</maven.compiler.source>
12            <maven.compiler.target>1.8</maven.compiler.target>
13          </properties>
```

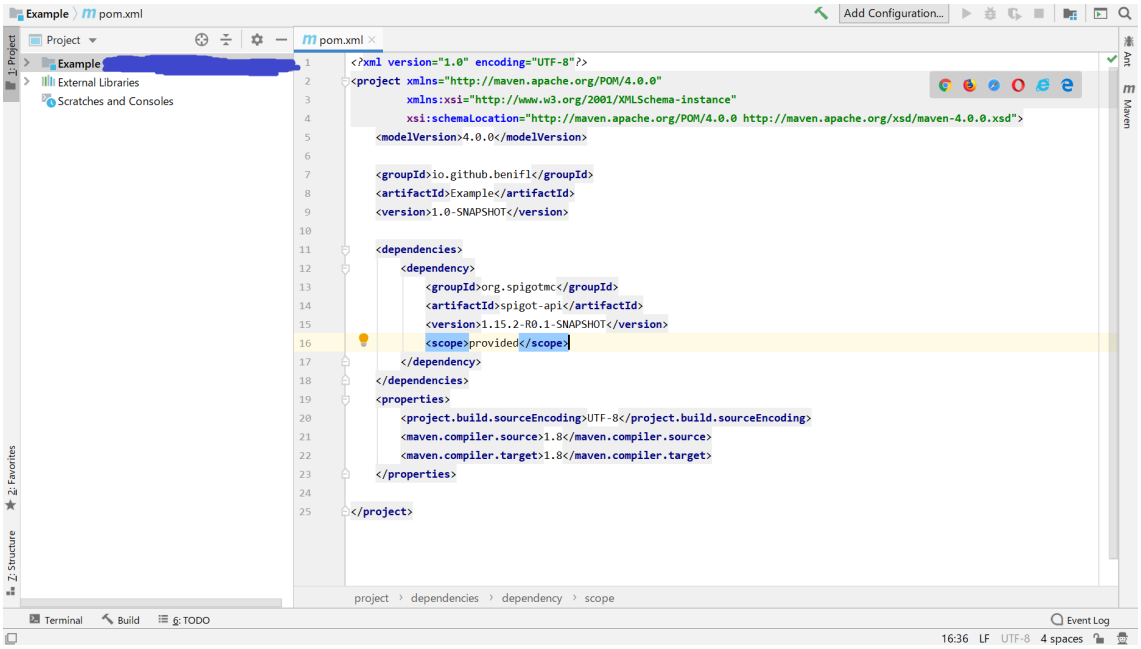

図 3　このようになっていれば OK

そして、pom.xml を右クリックし、”Maven → Reimport”を選択すれば準備は完了です。

4.2　最初のプラグイン作成

それでは、実際にプラグインを作成していきましょう。”src/main/java”にパッケージを作成し、そこに Java Class を作成します。

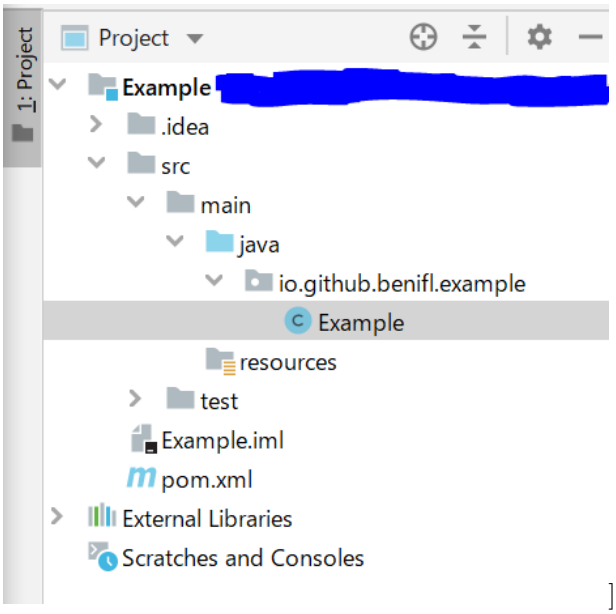

図 4

そして、以下のようにコードを記述してみましょう。

```
1  import org.bukkit.plugin.java.JavaPlugin;
2
3  public class Example extends JavaPlugin {
4    public void onEnable(){
5      getLogger().info("Hello World!");
6    }
7  }
```

次に、src/main/resources に”plugin.yml”を作成し、以下のように記述してみましょう。

```
1  name: Example
2  main: io.github.benifl.example.Example
3  version: 1.0.0
```

項目”name”にはプラグインの名前、”main”にはメインクラス（JavaPlugin クラスを継承するクラス）の場所（パッケージ名. クラス名）、”version”にはバージョンの数字（なんでもいい）を入力してください。

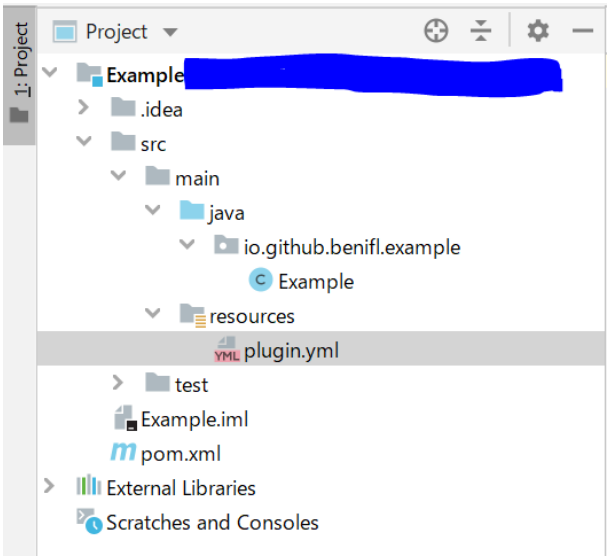

図 5　フォルダの構成はこのようになる

そして、画面右側の Maven タブを開き、”Lifecycle/package”をダブルクリックで実行します。

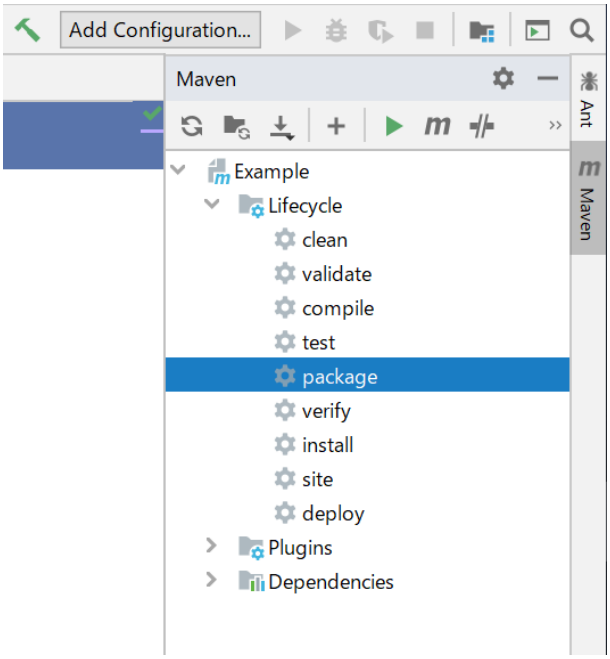

図 6

すると、新たに target フォルダが生成されました。この中にある jar ファイルがプラグインです！

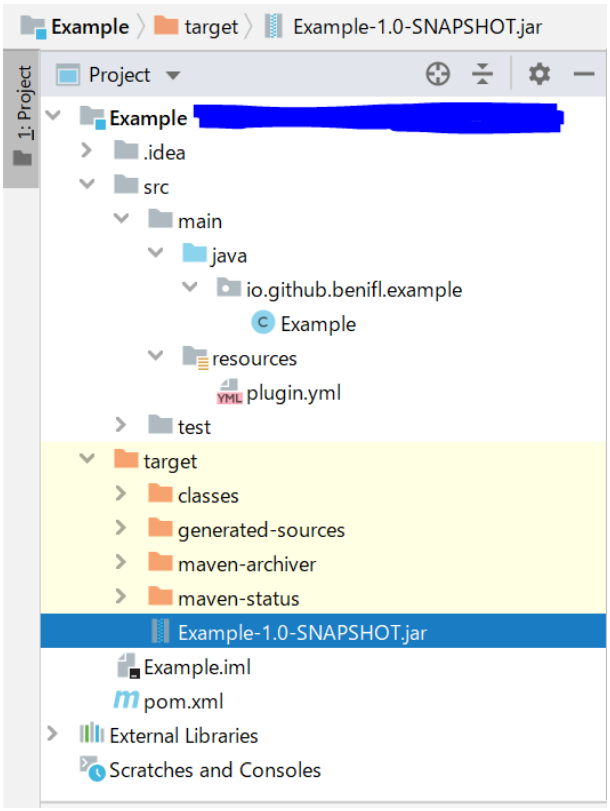

図 7

この jar ファイルを、前セクションで作成したサーバのフォルダ内の plugins フォルダにコピー&ペーストして、サーバを起動してみると…？

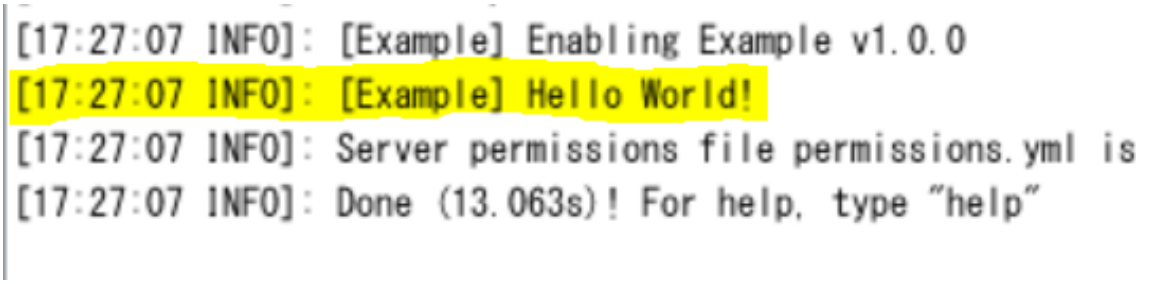

図 8

このように表示されました！ JavaPlugin#onEnable は、プラグインの起動時（サーバの起動時）に初めに実行されるメソッドです。プラグインの初期化処理はここに記述すると良いでしょう。

4.3 色々なものを作る:イベント処理

ここでは、プレイヤーがメインハンドに棒を持っているときに左クリックをするとカーソルの先に雷を落とすというプラグインを作ってみることにします。

ソースコード 1　Example.java

```
1  import org.bukkit.plugin.java.JavaPlugin;
2
3  public class Example extends JavaPlugin{
4      public void onEnable(){
5          //イベントリスナの登録
6          getServer().getPluginManager().registerEvents(new Event(),this);
7      }
8  }
```

ソースコード 2　Event.java

```
1  import org.bukkit.Material;
2  import org.bukkit.block.Block;
3  import org.bukkit.entity.Player;
4  import org.bukkit.event.EventHandler;
5  import org.bukkit.event.Listener;
6  import org.bukkit.event.block.Action;
7  import org.bukkit.event.player.PlayerInteractEvent;
8  import org.bukkit.util.BlockIterator;
9
10 public class Event implements Listener {
11     //プレイヤーがクリックをしたときに以下のメソッドが実行される
12     @EventHandler
13     public void onClick(PlayerInteractEvent event){
14         //左クリックかどうか
15         if(event.getAction().equals(Action.LEFT_CLICK_AIR)
16                 ||event.getAction().equals(Action.LEFT_CLICK_BLOCK)) {
17             Player player = event.getPlayer();
18             //プレイヤーのメインハンドのアイテムが棒かどうか
19             if (player.getInventory().getItemInMainHand()
20                     .getType().equals(Material.STICK)) {
21                 BlockIterator iterator = new BlockIterator(player);
22                 //100m先までプレイヤーの視線上のブロックを走査する
23                 for (int i = 0; i < 100; i++) {
24                     if (iterator.hasNext()) {
25                         Block block = iterator.next();
26                         //ブロックが空気以外であれば
27                         if (!block.getType().equals(Material.AIR)) {
28                             //ブロックの座標に雷を落とす
29                             block.getWorld().strikeLightning(block.getLocation());
30                             break;
31                         }
32                     } else {
33                         break;
34                     }
35                 }
36             }
37         }
```

```
38          }
39  }
```

図 9　実行結果

5　おわりに

ここまで読んでくださりありがとうございます。開発環境の導入という目標は達成されたのではないかと自負しておりますが、プログラミングの各論的な部分に関してこの資料だけでは説明が不十分なところもあるかと思います。もし困った場合は、まず「はじめに」に示した文献を見ていただければ、と考えております。特に、Spigot の JavaDoc の方は是非ともご参照ください。開発のヒントがたくさん詰まっています。

最後に、この資料が Minecraft で Java プログラミングを学ぶ際の一助となれば幸いです。