Creating A New Fragment Generator Java

We made it easy to create new custom fragment generators

Create a new class in your project and have it extend

```
FragmentGenerator
```

Once that is done, you should implement its methods

```java
public final class ExampleGenerator extends FragmentGenerator {

    public ExampleGenerator() {
        super("CustomGen", "example");
    }

    @Override
    public void handleGeneration(@NotNull Player player, @NotNull SerializedMap map) {

    }
}
```

After the class is configured to your satisfaction, you must add it to our generator list

```java
ISetsAPI.getFragmentGeneratorManager().registerFragmentGenerator(new ExampleGenerator());
```

Example Fishing Generator

We're not done yet; here's an example of how to make a listener that listens to the desired event for which you want the generator to be created.

```java
package com.insurgencedev.fishinggenerator;

import lombok.Getter;
import org.bukkit.entity.Player;
import org.insurgencedev.insurgencesets.api.FragmentGenerator;
import org.insurgencedev.insurgencesets.api.ISetsAPI;
import org.insurgencedev.insurgencesets.libs.fo.Common;
import org.insurgencedev.insurgencesets.libs.fo.collection.SerializedMap;
import org.insurgencedev.insurgencesets.models.armorset.ArmorSet;
import org.insurgencedev.insurgencesets.models.fragment.Fragment;
import org.insurgencedev.insurgencesets.settings.ISetsPlayerCache;
import org.jetbrains.annotations.NotNull;

@Getter
public final class FishingGenerator extends FragmentGenerator {

    public static final String namespace = "IGen";

    public FishingGenerator() {
        super(namespace, "Fishing");
    }

    @Override
    public void handleGeneration(@NotNull Player player, @NotNull SerializedMap map) {
        if (map.getBoolean("Enabled") && Math.random() <= map.getDouble("Chance") / 100) {
            ISetsPlayerCache cache = ISetsAPI.getCache(player);
            ArmorSet armorSet =
ISetsAPI.getArmorSetManager().findArmorSet(cache.getArmorSetFragmentGen());

            if (armorSet != null) {
                Fragment fragment = armorSet.getFragment();
                int amount = getAmount(map);

                if (map.getBoolean("Physical")) {
                    fragment.giveOrUpdateFragment(player, amount, false);
                } else {
                    cache.updateFragmentAmount(cache.getArmorSetFragmentGen(), amount);
                }
```

```
                Common.tellNoPrefix(player, map.getString("Give_Message").replace("{amount}",
"" + amount));
            }
        }
    }


    private int getAmount(SerializedMap map) {
        return map.getBoolean("Dynamic_Amount", false) &&
GeneratorAddon.getConfig().getFragmentAmount(PlayerFishingListener.getCaughtFish()) != null
                ?
GeneratorAddon.getConfig().getFragmentAmount(PlayerFishingListener.getCaughtFish())
                : map.getInteger("Amount_To_Give");
    }
}
```

The listener

```
package com.insurgencedev.fishinggenerator;

import lombok.Getter;
import org.bukkit.Material;
import org.bukkit.entity.Item;
import org.bukkit.entity.Player;
import org.bukkit.event.EventHandler;
import org.bukkit.event.Listener;
import org.bukkit.event.player.PlayerFishEvent;
import org.bukkit.inventory.ItemStack;
import org.insurgencedev.insurgencesets.api.ISetsAPI;
import org.insurgencedev.insurgencesets.models.armorset.ArmorSet;

import java.util.Arrays;
import java.util.List;

public final class PlayerFishingListener implements Listener {

    @Getter
    private static String caughtFish;

    @EventHandler
```

```java
    public void onFish(PlayerFishEvent event) {
        if (event.getState().equals(PlayerFishEvent.State.CAUGHT_FISH) && event.getCaught()
instanceof Item) {
            ItemStack item = ((Item) event.getCaught()).getItemStack();

            if (isFish(item.getType())) {
                Player player = event.getPlayer();
                ArmorSet armorSet =
ISetsAPI.getArmorSetManager().findArmorSet(ISetsAPI.getCache(player).getArmorSetFragmentGen())
;

                if (armorSet != null &&
FishingGenerator.namespace.equals(armorSet.getFragmentGeneration().getString("Type"))) {
                    List<String> disabledWorlds =
armorSet.getFragmentGeneration().getStringList("Disabled_Worlds");
                    if (!disabledWorlds.contains(player.getWorld().getName())) {
                        caughtFish = item.getType().name().toLowerCase();

ISetsAPI.getFragmentGeneratorManager().findFragmentGenerator(FishingGenerator.namespace,
armorSet.getFragmentGeneration().getString("Source"))
                                .handleGeneration(player, armorSet.getFragmentGeneration());
                    }
                }
            }
        }
    }

    private boolean isFish(Material material) {
        return Arrays.asList(Material.COD, Material.SALMON, Material.PUFFERFISH,
Material.TROPICAL_FISH).contains(material);
    }
}
```