InsurgenceBoosters

WIP

Setup

Get yourself started with the plugin!

Installing The Plugin

After downloading the plugin from BuiltByBit, place it in the plugins folder on your server. That's it!

Creating Your First Booster

The command below can be used to generate a booster. It will produce an automatically generated booster file that you can customize to your preference!

> /iboosters create <type> <namespace>
>
> Type: Can be anything that describes the kind of booster it will be. Take experience and money as examples.
> Namespace: Anything that uniquely distinguishes it from other boosters. Take MY_BOOSTER and VANILLA as examples.

Following the execution of that command, a new file should be visible in the /plugins/IBoosters/boosters folder.

Addon Roundup

What are addons?

Addons are used to extend the capabilities of the main plugin. We offer free addons which provide support for other plugins and also add additional features to the main plugin. You can view the entire list of addons on BuiltByBit. There are two types of addons, those that add support to plugins and those that add features to the main plugin. Plugin support addons will mostly be a `.Lua` file. They can also be `.jar` files depending on the complexity.

Below is a list of plugins we currently support. We are open to suggestions.

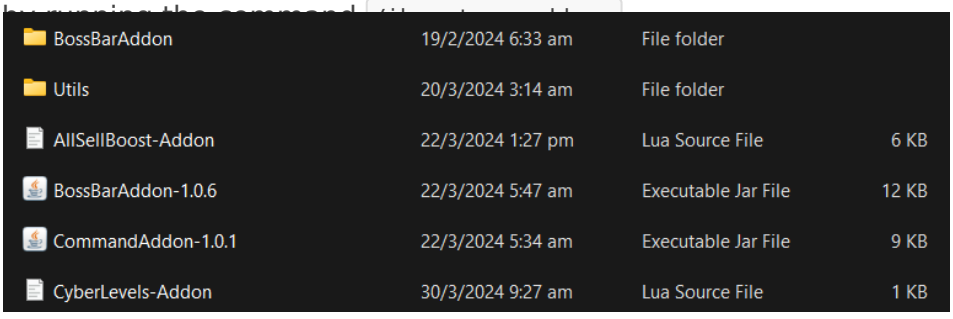
Supported Plugins

Download the Lua file for the plugin you wish to support, place it in the addons folder and reload the plugin!

ChunkCollector `Money`
DeluxeSellwands `Money`
EconomyShopGui `Money`
ManifestCollector `Money`
VirtualSpawner `Money`
WildChests `Money`
WildTools `Money`
ZithiumCollector `Money`
Sellwand `Money`
ShopGUIPlus `Money`
RivalFishingRods `Money, Essence, Tool XP`
RivalHarvesterHoes `Money, Essence, Tool XP`
RivalMobSwords `Money, Essence, Tool XP`
RivalPickaxes `Money, Essence, Tool XP`
JobsReborn `Money, XP, Points`
AlonosoLevels `Levels`
CyberLevels `Levels`
AuraSkills `Skills`
EcoSkills `Skills`
OptimalSkills `Skills`
mcMMO `Skills`
MMOCore `Skills`
BattlePass `Quest`
EcoQuests `Quest`
FlareMobCoins `Mobcoins`
SuperMobcoins `Mobcoins`
EcoJobs `Job XP`
EcoPets `Pet XP`
EdPrison `Currency & Enchant`
LevelTools `Tool XP`

Loading an addon

For an addon to be loaded, the `.jar` or `.Lua` file must be placed in the `plugins\IBoosters\addons` folder followed by the execution of the plugin's reload command or a server restart. You can verify if an addon has been loaded by running the command



Implementing an addon

Feature addons will work once loaded; no extra steps required. Plugin support addons are identified by a **Type** and a **Namespace**. You should be able to open the `.Lua` files with any given text editor and retrieve the type and namespace which is usually located at the top. See below example.

```
name = "Sellwand-Addon"
version = "1.0.0"
author = "Hxtch"
description = {"This addon will add", "Support for Sellwand boosting"}

local TYPE = "Sell"□□□   <<----------------------
local NAMESPACE = "SELLWAND"   <<----------------------

utils.subscribeToEvent("me.zachary.sellwand.api.events.SellwandSellEvent", function(event)
    local multi = boosterUtils.getMulti(event:getPlayer(), TYPE, NAMESPACE)

    if multi > 0 then
        event:setSellPrice(boosterUtils.calculateAmount(event:getSellPrice(), multi))
    end
end)
```

You will then be required to use the type and namespace in a booster's `YAML` file and that's it! See below example.

```
Type: Sell           <<----------------------
Namespace: SELLWAND  <<----------------------
Type_Display_Name: "Money"
Type_Menu_Item:
  Icon: GOLD_INGOT
  Display_Name: "&c&lMoney Booster"
  Priority: 0
  Model_Data: 0
  Lore:
    - ""
    - "&7&oThis boosts the amount"
    - "&7&o of money you earn"
```

```
    - ""
    - "&7Amount: &f{amount}"
    - ""
    - "&fClick &7to view your boosters"
Global_Item:
  Icon: GOLD_INGOT
  Model_Data: 0
  Display_Name: "&f&lGlobal &c&lMoney Booster"
  Glow_When_Active: false
  Inactive_Lore:
    - ""
    - "&7&oThis boosts the amount"
    - "&7&o of money you earn"
    - ""
    - "&7Time: &f{time}"
    - "&7Multiplier: &fx{multiplier}"
    - ""
    - "&fClick &7to activate"
  Activated_Lore:
    - ""
    - "&7&oThis boosts the amount"
    - "&7&o of money you earn"
    - ""
    - "&7Time: &f{time}"
    - "&7Multiplier: &fx{multiplier}"
    - "&7Remaining time: &f{remaining}"
Personal_Item:
  Icon: GOLD_INGOT
  Model_Data: 0
  Display_Name: "&f&lPersonal &c&lMoney Booster"
  Glow_When_Active: false
  Inactive_Lore:
    - ""
    - "&7&oThis boosts the amount"
    - "&7&o of money you earn"
    - ""
    - "&7Time: &f{time}"
    - "&7Multiplier: &fx{multiplier}"
    - ""
    - "&fClick &7to activate"
```

```
  Activated_Lore:
    - ""
    - "&7&oThis boosts the amount"
    - "&7&o of money you earn"
    - ""
    - "&7Time: &f{time}"
    - "&7Multiplier: &fx{multiplier}"
    - "&7Remaining time: &f{remaining}"
Menu:
  Title: "Money Boosters"
  Size: 45
  Booster_Slots: [ 10, 11, 12, 13, 14, 15, 16, 19, 20, 21, 22, 23, 24, 25 ]
  Inactive_Booster_Button:
    Icon: PAPER
    Display_Name: "&d&lActive Boosters"
    Slot: 40
    Model_Data: 0
    Lore:
      - ""
      - "&d▏ &7None"
      - ""
      - "&7➥ Activate a booster"
  Deactivate_Booster_Button:
    Icon: BARRIER
    Display_Name: "&c&lDeactivate"
    Slot: 41
    Model_Data: 0
    Lore:
      - ""
      - "&7➥ Deactivate the booster"
  Previous_Page_Button:
    Slot: 9
    Icon_Script: |-
      if canGo then
        return "LIME_DYE"
      else
        return "GRAY_DYE"
      end
    Model_Data: 0
  Next_Page_Button:
```

```
    Slot: 17
    Icon_Script: |-
      if canGo then
        return "LIME_DYE"
      else
        return "GRAY_DYE"
      end
    Model_Data: 0
  Misc_Items:
    - Display_Name: "&7"
      Slots: [ 0, 1, 2, 3, 4, 5, 6, 7, 8, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36, 37, 38, 39,
41, 42, 43, 44 ]
      Icon: BLACK_STAINED_GLASS_PANE
      Lore: []
      Glow: false
      Hide_Attributes: true
      Model_Data: 0
    - Display_Name: "&7"
      Slots: [ 18, 26 ]
      Icon: GRAY_STAINED_GLASS_PANE
      Lore: []
      Glow: false
      Hide_Attributes: true
      Model_Data: 0
```

Messages

The list of actions that can be taken on messages. Hex colors are supported in all messages.

Prefix

Insert the Prefix from settings.yml into any message just add `{prefix}` to the message

Json

Minecraft allows you to create more advanced messages using Json you can use a json text generator like this one here

To use json in a message all you have to do is put [JSON] at the start of the message

Example:

```
"[JSON][“”,{"text":"Example ","color":"dark_red"},{"text":"message
","color":"#E124E1"},{"text":"with ","color":"dark_gray"},{"text":"hover
","color":"gold","hoverEvent":{"action":"show_text","contents":"Hover
text!"}},{"text":"text","color":"red"}]"
```

Player Variable

You can also show the players name by inserting `{player}` into the message

Send Message As Action Bar

You can choose to send a chat message as an actionbar instead by adding `<actionbar>` to the start of the message

Send Message As Toast

You can choose to send a chat message as a toast by adding `<toast>` to the start of the message

Send Message As A Title

You can choose to send a chat message as a title by adding `<title>` to the start of the message

Send Message As A Boss Bar

You can choose to send a chat message as a bossbar by adding `<bossbar>` to the start of the message, it will show a boss bar for 10 seconds.

Center A Message In Chat

To center a message in chat add `<center>` to the start of the message.

Developer API

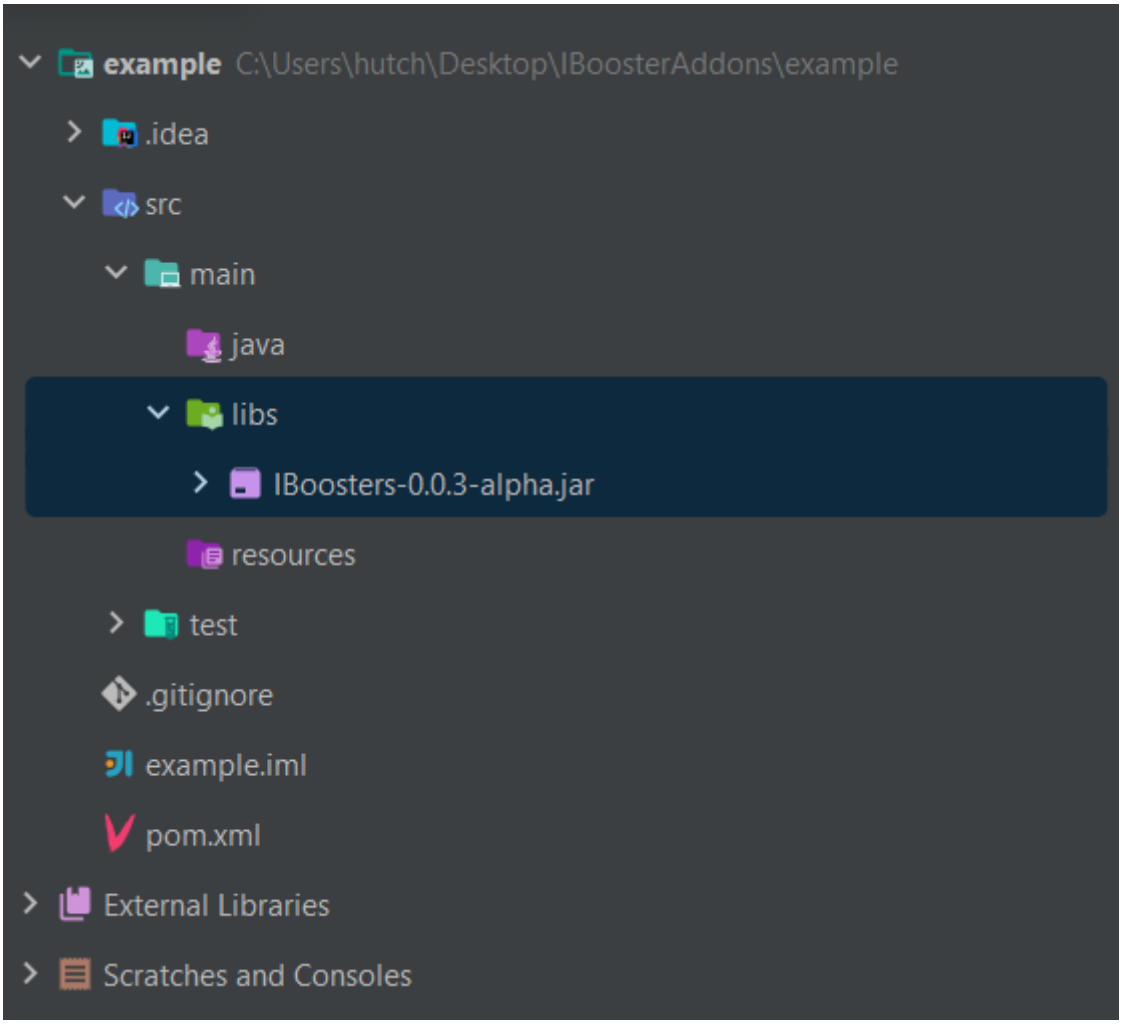
Use our robust API to put your unique ideas into practice.

Creating an Addon

Assuming you have already created your empty project, you will be required to create a new directory

called `libs` by right-clicking your `main` folder and clicking `New -> Directory`.

Cope and paste your IBoosters JAR file into that new folder.



Adding InsurgenceBoosters as a dependency

Add the below dependencies to your pom.xml file; replace `{JAR-NAME}` with the name of your IBoosters JAR file. A reference to the Spigot API is also included.

```
    <dependencies>
        <dependency>
            <groupId>org.insurgencedev</groupId>
            <artifactId>IBoosters</artifactId>
            <version>LATEST</version>
            <scope>system</scope>
            <systemPath>${project.basedir}/src/main/libs/{JAR-NAME}</systemPath>
        </dependency>
        <dependency>
            <groupId>org.spigotmc</groupId>
            <artifactId>spigot-api</artifactId>
            <version>1.20.2-R0.1-SNAPSHOT</version>
        </dependency>
    </dependencies>
```

Setting up the main class

After creating your main class, it should extend `InsurgenceBoostersAddon` and override the necessary methods. The class should also be annotated with `@IboostersAddon`.

```
package com.insurgencedev.exampleaddon;

import org.insurgencedev.insurgenceboosters.api.addon.IBoostersAddon;
import org.insurgencedev.insurgenceboosters.api.addon.InsurgenceBoostersAddon;

@IBoostersAddon(name = "ExampleAddon", version = "1.0.0", author = "Insurgence Dev Team",
description = "Example Addon")
public class ExampleAddon extends InsurgenceBoostersAddon {

    @Override
    public void onAddonStart() {
    }

    @Override
    public void onAddonReloadablesStart() {
    }

    @Override
```

```
    public void onAddonReload() {
    }

    @Override
    public void onAddonStop() {
    }

}
```

Creating a custom config

Make a fresh class. It should extend the `AddonConfig` class and override the `onLoad()` function. To load and generate your configuration file based on the file in your internal resource folder, make sure the `loadAddonConfig()` method is used in the constructor.

```
package org.insurgencedev.exampleaddon;

import org.insurgencedev.insurgencesets.api.addon.AddonConfig;

public class CustomConfig extends AddonConfig {

    public static String MYSTRING = null;



    public CustomConfig() {
        loadAddonConfig("config.yml", "config.yml");
    }

    @Override
    protected void onLoad() {
        MYSTRING = getString("Test");
    }
}
```

Creating a Custom Booster (Java)

Below is an example of an Experience booster. When a player receives exp, the amount received will be multiplied based on the boosters that are active. We locate active personal and global boosters of the required `Type` and `Namespace`. If located, we apply the associated multipliers.

```
package com.insurgencedev.exampleaddon.listeners;

import org.bukkit.event.EventHandler;
import org.bukkit.event.Listener;
import org.bukkit.event.player.PlayerExpChangeEvent;
import org.insurgencedev.insurgenceboosters.api.IBoosterAPI;
import org.insurgencedev.insurgenceboosters.models.booster.GlobalBoosterManager;
import org.insurgencedev.insurgenceboosters.settings.IBoostersPlayerCache;

public final class ExperienceChangeListener implements Listener {

    @EventHandler
    private void onChange(PlayerExpChangeEvent event) {
        final String TYPE = "Experience";
        final String NAMESPACE = "CUSTOM";
        double totalMulti = 1;

        IBoostersPlayerCache.BoosterFindResult pResult =
IBoosterAPI.getCache(event.getPlayer()).findActiveBooster(TYPE, NAMESPACE);
        if (pResult instanceof IBoostersPlayerCache.BoosterFindResult.Success boosterResult) {
            totalMulti += boosterResult.getBooster().getMultiplier();
        }

        GlobalBoosterManager.BoosterFindResult gResult =
IBoosterAPI.getGlobalBoosterManager().findBooster(TYPE, NAMESPACE);
        if (gResult instanceof GlobalBoosterManager.BoosterFindResult.Success boosterResult) {
            totalMulti += boosterResult.getBooster().getMultiplier();
```

```
        }

        event.setAmount((int) (event.getAmount() * totalMulti));
    }
}
```

Register your listener

For your listener class to take effect, it must be registered.

```
package com.insurgencedev.exampleaddon;

import com.insurgencedev.exampleaddon.listeners.ExperienceChangeListener;
import org.insurgencedev.insurgenceboosters.api.addon.IBoostersAddon;
import org.insurgencedev.insurgenceboosters.api.addon.InsurgenceBoostersAddon;

@IBoostersAddon(name = "ExampleAddon", version = "1.0.0", author = "Insurgence Dev Team",
description = "Example Addon")
public class ExampleAddon extends InsurgenceBoostersAddon {

    @Override
    public void onAddonReloadablesStart() {
        registerEvent(new ExperienceChangeListener());
    }

}
```

Creating an Effect Booster (Java)

Below is an example of how you can create a booster that applies effects to the player

Main Class

```
package com.insurgencedev.potioneffectsaddon;

import com.insurgencedev.potioneffectsaddon.listeners.BoosterListener;
import org.insurgencedev.insurgenceboosters.api.addon.IBoostersAddon;
import org.insurgencedev.insurgenceboosters.api.addon.InsurgenceBoostersAddon;

@IBoostersAddon(name = "PotionEffects", version = "1.0.0", author = "Insurgence Dev Team",
description = "Apply Potion Effects")
public class PotionEffectsAddon extends InsurgenceBoostersAddon {

    @Override
    public void onAddonStart() {
        new MyConfig();
    }

    @Override
    public void onAddonReloadablesStart() {
        registerEvent(new BoosterListener());
    }

}
```

Listener Class

```
package com.insurgencedev.potioneffectsaddon.listeners;

import com.insurgencedev.potioneffectsaddon.MyConfig;
import org.bukkit.entity.Player;
import org.bukkit.event.EventHandler;
import org.bukkit.event.EventPriority;
import org.bukkit.event.Listener;
import org.bukkit.event.player.PlayerItemConsumeEvent;
import org.bukkit.event.player.PlayerJoinEvent;
import org.bukkit.event.player.PlayerRespawnEvent;
import org.bukkit.potion.PotionEffectType;
import org.insurgencedev.insurgenceboosters.api.IBoosterAPI;
import org.insurgencedev.insurgenceboosters.events.IBoosterEndEvent;
import org.insurgencedev.insurgenceboosters.events.IBoosterStartEvent;
import org.insurgencedev.insurgenceboosters.libs.fo.Common;
import org.insurgencedev.insurgenceboosters.libs.fo.remain.CompMaterial;
import org.insurgencedev.insurgenceboosters.models.booster.GlobalBoosterManager;
import org.insurgencedev.insurgenceboosters.settings.IBoostersPlayerCache;

import java.util.ArrayList;
import java.util.List;
import java.util.UUID;

public final class BoosterListener implements Listener {

    private final List<UUID> activeList = new ArrayList<>();
    private final String NAMESPACE = "CUSTOM_EFFECT";

    @EventHandler
    private void onStart(IBoosterStartEvent event) {
        if (event.getBoosterData().getType().equals(MyConfig.boosterType) &&
                event.getBoosterData().getNamespace().equals(NAMESPACE)) {

            Player player = event.getPlayer();

            if (MyConfig.disabledWorlds.contains(player.getWorld().getName())) {
                event.getBoosterData().setActive(false);
                return;
            }
```

```
            if (alreadyHasEffect(player)) {
                event.getBoosterData().setActive(false);
                return;
            }

            activeList.add(player.getUniqueId());
            applyEffect(player);
        }
    }

    @EventHandler
    private void onEnd(IBoosterEndEvent event) {
        if (event.getBoosterData().getType().equals(MyConfig.boosterType) &&
                event.getBoosterData().getNamespace().equals(NAMESPACE)) {

            Player player = event.getPlayer();

            activeList.remove(player.getUniqueId());
            removeEffect(player);
        }
    }

    @EventHandler
    private void onConsume(PlayerItemConsumeEvent event) {
        if (CompMaterial.fromItem(event.getItem()).equals(CompMaterial.MILK_BUCKET)) {
            Player player = event.getPlayer();

            IBoostersPlayerCache.BoosterFindResult pResult =
IBoosterAPI.getCache(player).findActiveBooster(MyConfig.boosterType, NAMESPACE);
            if (pResult instanceof IBoostersPlayerCache.BoosterFindResult.Success) {
                Common.runLater(2, () -> applyEffect(player));
                return;
            }

            GlobalBoosterManager.BoosterFindResult gResult =
IBoosterAPI.getGlobalBoosterManager().findBooster(MyConfig.boosterType, NAMESPACE);
            if (gResult instanceof GlobalBoosterManager.BoosterFindResult.Success) {
                Common.runLater(2, () -> applyEffect(player));
            }
        }
```

```
    }

    @EventHandler(priority = EventPriority.MONITOR)
    private void onRespawn(PlayerRespawnEvent event) {
        Player player = event.getPlayer();

        IBoostersPlayerCache.BoosterFindResult pResult =
IBoosterAPI.getCache(player).findActiveBooster(MyConfig.boosterType, NAMESPACE);
        if (pResult instanceof IBoostersPlayerCache.BoosterFindResult.Success) {
            Common.runLater(2, () -> applyEffect(player));
            return;
        }

        GlobalBoosterManager.BoosterFindResult gResult =
IBoosterAPI.getGlobalBoosterManager().findBooster(MyConfig.boosterType, NAMESPACE);
        if (gResult instanceof GlobalBoosterManager.BoosterFindResult.Success) {
            Common.runLater(2, () -> applyEffect(player));
        }
    }

    @EventHandler(priority = EventPriority.MONITOR)
    private void onJoin(PlayerJoinEvent event) {
        GlobalBoosterManager.BoosterFindResult gResult =
IBoosterAPI.getGlobalBoosterManager().findBooster(MyConfig.boosterType, NAMESPACE);
        if (gResult instanceof GlobalBoosterManager.BoosterFindResult.Success) {
            Player player = event.getPlayer();

            if (!activeList.contains(player.getUniqueId())) {
                activeList.add(player.getUniqueId());
                Common.runLater(2, () -> applyEffect(player));
            }
        }
    }

  @EventHandler(priority = EventPriority.MONITOR)
    private void onLeave(PlayerQuitEvent event) {
        Player player = event.getPlayer();

        IBoostersPlayerCache.BoosterFindResult pResult =
IBoosterAPI.getCache(player).findActiveBooster(MyConfig.boosterType, NAMESPACE);
```

```
        if (pResult instanceof IBoostersPlayerCache.BoosterFindResult.Success) {
            removeEffect(player);
            return;
        }

        GlobalBoosterManager.BoosterFindResult gResult =
IBoosterAPI.getGlobalBoosterManager().findBooster(MyConfig.boosterType, NAMESPACE);
        if (gResult instanceof GlobalBoosterManager.BoosterFindResult.Success) {
            removeEffect(player);
        }
    }

    @EventHandler(priority = EventPriority.MONITOR)
    private void onKick(PlayerKickEvent event) {
        Player player = event.getPlayer();

        IBoostersPlayerCache.BoosterFindResult pResult =
IBoosterAPI.getCache(player).findActiveBooster(MyConfig.boosterType, NAMESPACE);
        if (pResult instanceof IBoostersPlayerCache.BoosterFindResult.Success) {
            removeEffect(player);
            return;
        }

        GlobalBoosterManager.BoosterFindResult gResult =
IBoosterAPI.getGlobalBoosterManager().findBooster(MyConfig.boosterType, NAMESPACE);
        if (gResult instanceof GlobalBoosterManager.BoosterFindResult.Success) {
            removeEffect(player);
        }
    }

    private boolean alreadyHasEffect(Player player) {
        PotionEffectType type = PotionEffectType.getByName(MyConfig.effectNamespace);
        return type != null && player.hasPotionEffect(type);
    }

    private void applyEffect(Player player) {
        PotionEffectType type = PotionEffectType.getByName(MyConfig.effectNamespace);
        if (type != null) {
            player.addPotionEffect(type.createEffect(Integer.MAX_VALUE,
MyConfig.effectAmplifier - 1));
```

```
        }
    }

    private void removeEffect(Player player) {
        PotionEffectType type = PotionEffectType.getByName(MyConfig.effectNamespace);
        if (type != null) {
            player.removePotionEffect(type);
        }
    }
}
```

Custom Config Class

```
package com.insurgencedev.potioneffectsaddon;

import lombok.Getter;
import org.insurgencedev.insurgenceboosters.api.addon.AddonConfig;

import java.util.List;

@Getter
public class MyConfig extends AddonConfig {

    public static String boosterType;
    public static String effectNamespace;
    public static int effectAmplifier;
    public static List<String> disabledWorlds;

    public MyConfig() {
        loadAddonConfig("config.yml", "config.yml");
    }

    @Override
    protected void onLoad() {
        boosterType = getString("Type");
        effectNamespace = getString("Effect");
        effectAmplifier = getInteger("Amplifier");
        disabledWorlds = getStringList("Disabled_Worlds");
```

```
    }
}
```

Internal Config file

```
Type: Strength
Effect: INCREASE_DAMAGE
Amplifier: 2
Disabled_Worlds:
  - example
```

Placeholders

PlaceholderAPI

Below is a list of placeholders that can be used to retrieve data about the boosters.

> {scope} - Booster scope (Personal or Global)
> {type} - Name of the booster (Money, Token)
> {namespace} - Unique identifier for the booster (CUSTOM, VANILLA)

Get how much time is left on an active booster

```
%iboosters_{scope}_{type}_{namespace}_time_left%
```

Get the multiplier of an active booster

```
%iboosters_{scope}_{type}_{namespace}_multiplier%
```

Get the owner of an active global booster

```
%iboosters_global_{type}_{namespace}_owner%
```

Returns whether or not a booster is active

```
%iboosters_{scope}_{type}_{namespace}_is_active%
```

Returns a permanent booster

```
%iboosters_{scope}_{type}_{namespace}_permanent%
```