

Developer API

This guide assumes you already know how Java works. If you have any suggestions or questions about the API then reach out to us through Discord or any other contact form

- [Creating Addon Lua](#)
- [Creating A Custom Fragment Generator Lua](#)
- [Creating An Addon](#)
- [Creating A New Fragment Generator Java](#)
- [Creating A New Currency Java](#)

Creating Addon Lua

To accommodate addons that necessitate only a minimal amount of code, we have introduced the option to utilize Lua in place of Java. The process of establishing a Lua addon is straightforward and user-friendly.

Setting up the addon

Addons require these fields or else it will fail to load

```
name = "Example- Addon"
version = "1.0.0"
author = "InsurgenceDev"
description = {
    "This is an example lua addon"
}
```

Printing text to console when addon is started

To print text to the console when the addon is started you need to create a function named "onAddonStart" really its the same as Java addons

```
name = "Example- Addon"
version = "1.0.0"
author = "InsurgenceDev"
description = {
    "This is an example lua addon"
}

function onAddonStart()
    print("Lua addon started")
end
```

```
end
```

Importing classes from java

to import a class is very simple

```
import ' java.util. ArrayList'
```

Above will import the ArrayList java class example of how to use it

```
import ' java.util. ArrayList'

local arrayList = ArrayList()

function onAddonStart()
  arrayList: add(' Example 1' )
  arrayList: add(' Example 2' )
  arrayList: add(' Example 3' )
  arrayList: add(' Example 4' )
end
```

Utils

There is a utils package as well you can find it [here](#)

Creating A Custom Fragment Generator Lua

It is possible to use Lua to develop a simple or new fragment generator for additional events.

First create a new folder in `/Isets/` named fragment-generators should be like `/Isets/fragment-generators/`. Once that is done, you should create a new file. You can name it whatever you want. For this guide, I will name it `example-generator.lua`.

Creating the generator file

We will create a generator for the `BlockBreakEvent`. It will be similar to the default generator provided by the plugin.

```
package.path = pluginFolder .. "\\fragment-generators\\listeners\\?.lua"

namespace = "example"
source = "blockBreakEvent"
require "BlockBreakEvent"

function handleGeneration(player, settingsMap)
    if settingsMap:getBoolean("Enabled") and math.random() <= settingsMap:getDouble("Chance")
    then
        local amount = (settingsMap:getDouble("Amount_To_Give") <= 0) and 1 or
settingsMap:getDouble("Amount_To_Give")
        local fragment = utils.findFragment(player)
        if fragment ~= nil then
            if settingsMap:getBoolean("Physical") then
                givePhys(player, amount, fragment)
            else
                giveVirtual(player, amount, fragment)
            end
        end
    end
end
```

```

end
end

end

function givePhys(player, amount, fragment)
    fragment: giveOrUpdateFragment(player, math.floor(amount), false)
    utils.tell(player, "&2You been given " .. amount .. " &6fragments &2for &6" ..
fragment: getArmorSetName())
end

function giveVirtual(player, amount, fragment)
    local cache = utils.getCache(player)
    local currentBalance = cache: getFragmentAmount( fragment: getArmorSetName())
    cache: updateFragmentAmount( fragment. armorSetName, currentVirtualBalance +
math.floor(amount))
    utils.tell(player, "&2You been given " .. amount .. " &6fragments &2for &6" ..
fragment: getArmorSetName())
end

```

Registering an event listener

To actually create a listener for the `BlockBreakEvent`, you should first create a folder inside `/Isets/`. In this example, we are name the folder 'listeners'; `/Isets/fragment-generators/listeners/`. Create a new file and in this case, we name it `BlockBreak.lua`.

```

utils.subscribeToEvent("org. bukkit. event. block. BlockBreakEvent", function(event)
    local player = event:getPlayer()
    local armorSet =
utils.findArmorSet(utils.getCache(player): getFragmentDataManager(): getArmorSetFragmentGen())

    if armorSet ~= nil then
        local type = armorSet: getFragmentGeneration(): getString("Type")
        local source = armorSet: getFragmentGeneration(): getString("Source")
        local fragmentGenerator = utils.findFragmentGenerator(type, source)

        if fragmentGenerator == nil then
            return
        end
    end
end

```

```

if type == namespace then
  local disabledWorlds = armorSet:getFragmentGeneration():getStringList("Disabled_Worlds")
  local worldName = player:getWorld():getName()
  for i = 0, disabledWorlds:size() - 1 do
    if disabledWorlds:get(i) == worldName then
      return
    end
  end
  fragmentGenerator:handleGeneration(player, armorSet:getFragmentGeneration())
end
end)

```

Utils package

The utils package has a few utility functions.

Util to register an event listener.

```

subscribeToEvent("path to event class", function(event)
  -- Code here
end)

```

Util to retrieve an armor set. If none was located, the function will return nil.

```

findArmorSet("name of the armor set")

```

Util to retrieve a fragment generator. If no fragment generator was discovered, the function will return nil.

```

findFragmentGenerator("type/namespace", "source")

```

Util to retrieve a player's cache. It will return nil if the cache cannot be located; if it can, it will return the cache.

```
getCache(player)
```

Use this to determine which fragment generator a player currently has enabled. If "none" is returned, then they did not enable one. The name of the armor set for which they activated the generator for will be returned.

```
findFragment(player)
```

Util to send a message to a player. It will auto translate colors.

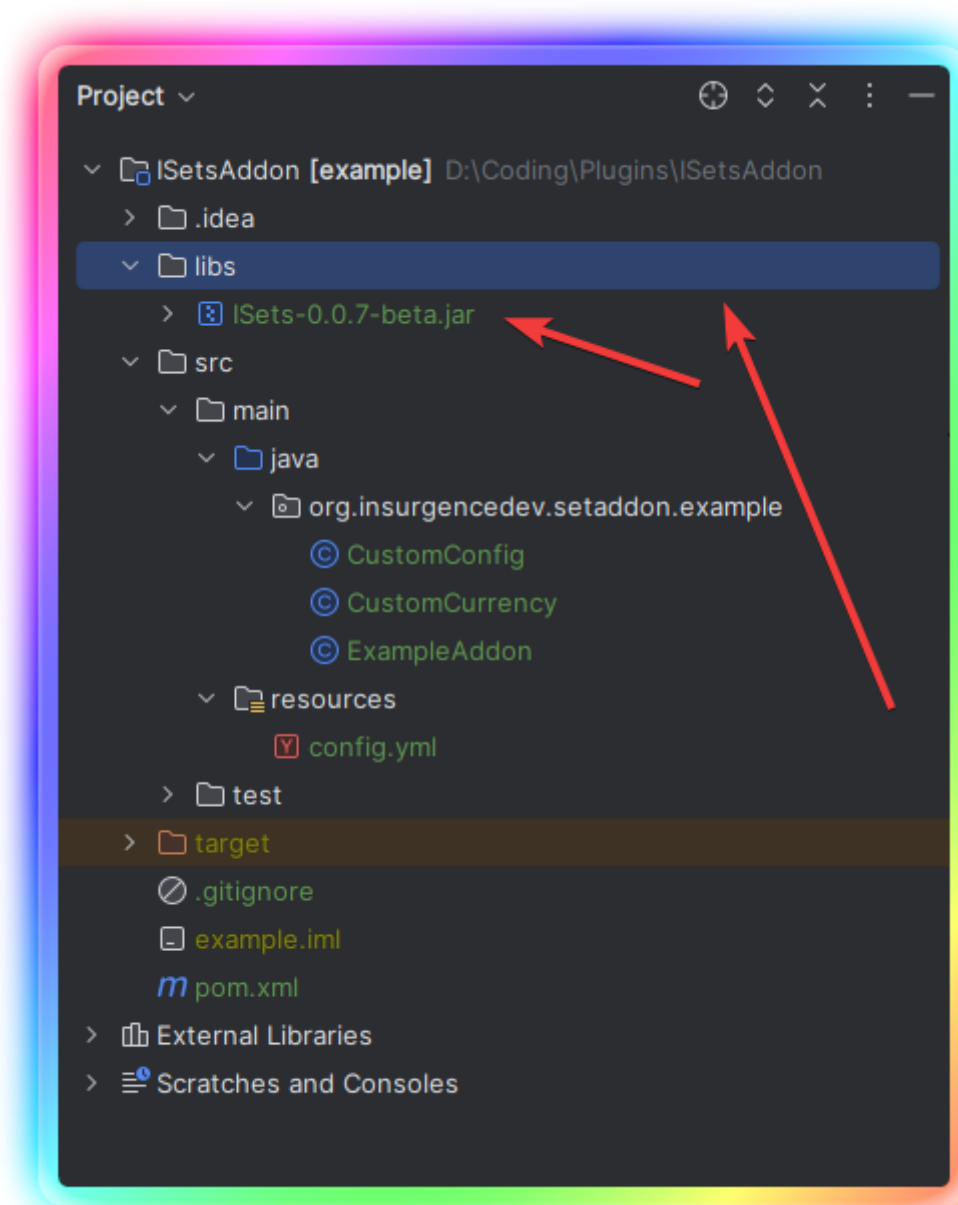
```
tell(player, "message goes here color codes and hex is automaticly translated")
```

Creating An Addon

Setup a new project then right click the project and create a directory called libs.

Once that is done drag in the latest version of ISets

Download example project [here](#)



Setting up the pom.xml

You will have to add a dependency pointing to ISets in the libs folder, and you will also need to add a dependency for the spigot api

```
<dependency>
  <groupId>org. insurgedev</groupId>
  <artifactId>insurgencesets</artifactId>
  <version>LATEST</version>
  <scope>system</scope>
  <systemPath>${project.basedir}/libs/ISets-0.0.7-beta.jar</systemPath>
</dependency>
<dependency>
  <groupId>org. spigotmc</groupId>
  <artifactId>spigot-api</artifactId>
  <version>1.20.2-R0.1-SNAPSHOT</version>
</dependency>
```

Setting up the main class

```
package org. insurgedev. setaddon. example;

import org. insurgedev. insurgencesets. api. addon. ISetsAddon;
import org. insurgedev. insurgencesets. api. addon. InsurgenceSetsAddon;
import org. insurgedev. insurgencesets. libs. fo. Common;
import org. insurgedev. insurgencesets. models. currency. CurrencyManager;

@ISetsAddon(name = "ExampleAddon", version = "1.0.0", author = "Insurgence Dev Team",
description = {"This is a test", "addon it serves no purpose", "other than for testing"})
public class ExampleAddon extends InsurgenceSetsAddon {

  @Override
  public void onAddonStart() {
  }

  @Override
  public void onAddonReloadablesStart() {
  }
```

```

@Override
public void onAddonReload() {
}

@Override
public void onAddonStop() {
}

}

```

Creating a config for your addon

Create a new class and name it whatever you want, then extend `AddonConfig`.

In the constructor, use `loadAddonConfig()`. First argument is the location to the default config located in resources.

```

package org.insurgencedev.setaddon.example;

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

public class CustomConfig extends AddonConfig {

    public static String TEST_STRING = null;

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

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

}

```

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

```
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

```
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.

```

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

```

```

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);
    }
}

```

Creating A New Currency Java

Creating a new currency is very easy; simply extend our Currency class

Currency

After extending the class, ensure to implement its methods, resulting in a class structured as depicted below

```
package org.insurgencedev.mobcoins;

import lombok.NonNull;
import org.bukkit.entity.Player;
import org.insurgencedev.insurgencesets.api.ISetsAPI;
import org.insurgencedev.insurgencesets.api.currency.Currency;
import org.insurgencedev.insurgencesets.models.currency.TransactionTypes;

public class ExampleCurrency extends Currency {

    public ExampleCurrency() {
        super("example", "ex");
    }

    @Override
    public boolean canAfford(@NonNull Player player, @NonNull Object amount) {
        return true;
    }

    @NonNull
    @Override
    public TransactionTypes handleDeposit(@NonNull Player player, @NonNull Object amount,
String armorSetName) {
        return TransactionTypes.SUCCESS;
    }
}
```

```

        @NonNull
        @Override
        public TransactionTypes handleTransaction(@NonNull Player player, @NonNull Object amount,
String armorSetName) {
            return TransactionTypes.SUCCESS;
        }
    }
}

```

MobCoins Example

In the next example, we are going to make a currency for SuperMobCoins

```

package org.insurgencedev.mobcoins;

import lombok.NonNull;
import me.swanis.mobcoins.MobCoinsAPI;
import org.bukkit.entity.Player;
import org.insurgencedev.insurgencesets.api.ISetsAPI;
import org.insurgencedev.insurgencesets.api.currency.Currency;
import org.insurgencedev.insurgencesets.models.currency.TransactionTypes;

public class MobCoinCurrency extends Currency {

    public MobCoinCurrency() {
        super("MobCoins", "SM");
    }

    @Override
    public boolean canAfford(@NonNull Player player, @NonNull Object o) {
        return MobCoinsAPI.getProfileManager().getProfile(player).getMobCoins() >= ((Number)
o).longValue();
    }

    @NonNull
    @Override
    public TransactionTypes handleDeposit(@NonNull Player player, @NonNull Object o, String s)
{
        if (isValidSet(s)) {

```



```

        return TransactionTypes.FAIL;
    }

    MobCoinsAPI.getProfileManager().getProfile(player).setMobCoins(MobCoinsAPI.getProfileManager()
        .getProfile(player).getMobCoins() + ((Number) o).longValue());
    return TransactionTypes.SUCCESS;
}

@NonNull
@Override
public TransactionTypes handleTransaction(@NonNull Player player, @NonNull Object o,
String s) {
    if (isInvalidSet(s)) {
        return TransactionTypes.FAIL;
    }

    long amount = ((Number) o).longValue();
    if (MobCoinsAPI.getProfileManager().getProfile(player).getMobCoins() < amount) {
        return TransactionTypes.FAIL_INSUFFICIENT_FUNDS;
    }

    MobCoinsAPI.getProfileManager().getProfile(player).setMobCoins(MobCoinsAPI.getProfileManager()
        .getProfile(player).getMobCoins() - ((Number) o).longValue());
    return TransactionTypes.SUCCESS;
}

private boolean isInvalidSet(String armorSet) {
    return armorSet == null || ISetsAPI.getArmorSetManager().findArmorSet(armorSet) ==
null;
}
}

```

You can choose to let the armor sets increase your currency earnings. When your currency is provided to the player, it must trigger an event at which you can listen to and boost accordingly.

```

package org.insurgencedev.mobcoins;

import lombok.NonNull;

```

```

import me.swanis.mobcoins.MobCoinsAPI;
import me.swanis.mobcoins.events.MobCoinsReceiveEvent;
import org.bukkit.entity.Player;
import org.bukkit.event.EventHandler;
import org.bukkit.event.Listener;
import org.bukkit.inventory.ItemStack;
import org.insurgencedev.insurgencesets.api.ISetsAPI;
import org.insurgencedev.insurgencesets.api.currency.Currency;
import org.insurgencedev.insurgencesets.libs.fo.remain.nbt.NBTItem;
import org.insurgencedev.insurgencesets.models.armorset.ArmorSet;
import org.insurgencedev.insurgencesets.models.currency.TransactionTypes;
import org.insurgencedev.insurgencesets.models.upgrade.Boost;
import org.insurgencedev.insurgencesets.models.upgrade.Upgrade;
import org.insurgencedev.insurgencesets.settings.ArmorSetData;
import org.insurgencedev.insurgencesets.settings.ISetsPlayerCache;

public class MobCoinReceiveListener implements Listener {

    @EventHandler
    public void onEarn(MobCoinsReceiveEvent event) {
        if (!MobCoinsCurrencyAddon.isDependentEnabled()) {
            return;
        }

        Player player = event.getProfile().getPlayer();
        ISetsPlayerCache cache = ISetsPlayerCache.from(player);

        ItemStack[] armorContents = player.getInventory().getArmorContents();
        for (ItemStack item : armorContents) {
            if (item != null) {
                NBTItem nbtItem = new NBTItem(item);
                if (!nbtItem.hasTag("armorSet")) {
                    continue;
                }

                ArmorSet armorSet =
ISetsAPI.getArmorSetManager().findArmorSet(nbtItem.getString("armorSet"));
                if (armorSet == null) {
                    continue;
                }
            }
        }
    }
}

```

```

        String armorSetName = armorSet.getName();
        String itemType = item.getType().name().split("_")[1];
        ArmorSetData armorSetData = cache.getArmorSetData(armorSetName);
        if (armorSetData == null) {
            continue;
        }

        Object levels = getLevelsFromType(itemType, armorSetData);
        if (levels instanceof Integer) {
            Upgrade upgrade = armorSet.findPieceLevels(itemType, (Integer) levels);
            if (upgrade == null) {
                continue;
            }

            for (Boost boost : upgrade.getBoosts()) {
                if ("CURRENCY".equals(boost.getNamespace()) &&
boost.getType().equals("MobCoins")) {
                    double boostAmount =
boost.getBOOST_SETTINGS().getDouble("Boost_Amount");
                    event.setAmount(calcAmountToGive(event.getAmount(), boost,
boostAmount));
                }
            }
        }
    }
}

private long calcAmountToGive(long amountFromEvent, Boost boost, double boostAmount) {
    if (boost.isPercent()) {
        return (long) (amountFromEvent * (1 + boostAmount / 100));
    } else {
        return (long) (amountFromEvent * (boostAmount < 1 ? 1 + boostAmount :
boostAmount));
    }
}

private Object getLevelsFromType(String type, ArmorSetData armorSetData) {
    return switch (type) {

```

```

        case "HEAD", "HELMET" -> armorSetData.getHelmetLevels();
        case "CHESTPLATE" -> armorSetData.getChestplateLevels();
        case "LEGGINGS" -> armorSetData.getLeggingsLevels();
        case "BOOTS" -> armorSetData.getBootsLevels();
        default -> false;
    };
}
}

```

Main class

```

package org.insurgencedev.mobcoins;

import org.bukkit.Bukkit;
import org.insurgencedev.insurgencesets.api.ISetsAPI;
import org.insurgencedev.insurgencesets.api.addon.ISetsAddon;
import org.insurgencedev.insurgencesets.api.addon.InsurgenceSetsAddon;
import org.insurgencedev.insurgencesets.libs.fo.Common;

@ISetsAddon(name = "SM-Mobcoins", version = "1.0.0", author = "Insurgence Dev Team",
description = "Use SuperMobCoins's mobcoins as a currency")
public class MobCoinsCurrencyAddon extends InsurgenceSetsAddon {

    @Override
    public void onAddonStart() {
        if (isDependentEnabled()) {
            registerEvent(new MobCoinReceiveListener());
        }
    }

    @Override
    public void onAddonReloadablesStart() {
        if (isDependentEnabled()) {
            ISetsAPI.getCurrencyManager().registerCurrency(new MobCoinCurrency());
        }
    }

    public static boolean isDependentEnabled() {
        return Bukkit.getPluginManager().isPluginEnabled("SuperMobCoins");
    }
}

```

