# Создание плагина для MyBPM

1) Установите утилиту “gg”.

curl <https://raw.githubusercontent.com/greetgo/greetgo.cmd/master/scripts/install.bash> -sSf | bash

2) Создайте проект с помощью установленной утилиты. Введя команду gg появятся дальнейшие возможные продолжения команды. При каждом нажатии на enter вы увидите дальнейшие подсказки. Следуйте предлагаемой инструкции. Примечания:

* project\_name не должен содержать префикса mybpm-plugin. Например, если проект называется mybpm-plugin-jusan, то название на данном этапе вводите jusan.
* для создания плагина используйте тип mybpm-plugin-templates
* указываемый порт - это первый базовый порт, для остальных вещей (фронта, бэка mybpm, кафки и т.д.) будет использоваться базовый порт с последующей инкрементацией (пример: базовый порт: 5000, порт для кафки будет 5001, для монги будет 5002 и т.д.) При завершении команды ключевым словом go проект начнет генерироваться Итоговая команда: gg new project <project\_name> mybpm-plugin-templates main -port-base=<port\_number> go

**Описание архитектуры плагина и его разработки в процессе**

### Проблема dependency hell в плагине

#### Проблема:

На плагине сервис в public static void main(String[] args){}, или в тестах успешно работает. При подключении ядра и поднятия всего контекста - сервис перестает работать.

В первый раз при вызове сервиса падает с ошибкой (NoSuchMethodError):

"message": "Handler dispatch failed; nested exception is java.lang.NoSuchMethodError: 'okhttp3.MediaType okhttp3.MediaType.get(java.lang.String)'",

Все остальные запросы теряют определение класса в процессе рантайма (NoClassDefFoundError):

"message": "Handler dispatch failed; nested exception is java.lang.NoClassDefFoundError: Could not initialize class com.infobip.RequestFactory$MediaTypeRegistry",

#### Например:

* com.squareup.okhttp3:3.\* версии написан на Java;
* com.squareup.okhttp3:4.\* и выше переписали на Kotlin, изменили метод обращения к MediaType.

Чтобы посмотреть дерево зазивисмостей можно запустить:

./gradlew allDeps

где

(\*) - зависимость ранее встречалась
(n) - not resolved
(c) - dependency constraint

или точечно для либы:

./gradlew :plugin\_module\_name:dependencyInsight --dependency okhttp3

#### Решение:

**зайти в контейнер** плагина:

docker exec -it container\_name bash

тут есть две директории:

* /app/external - внешние .jar пакеты ядра (core mybpm);
* /app/plugins/mybpm-plugin-monamie/external - внешние .jar пакеты плагина (plugin mybpm).

Cпособ рабочий, но не рекомендуемый. Может не скомпилироваться ядро проекта. Точечно удалить .jar командой RUN в Dockerfile, создающем image плагина (и на его место поставить другую версию либы при необходимости):

RUN cd /app/extern \
&& rm /app/extern/okhttp-3.9.0.jar \
&& rm /app/extern/okio-1.13.0.jar

Идеальный вариант: обновить старую библиотеку на ядре проекта.

#### Дополнительно:

По приоритету gradle подтягивает первый обнаруженный пакет, поэтому в приоритете компилируется jar ядра проекта.

**Установка принудительной версии** (строгой зависимости) на плагине не поможет:

implementation("com.squareup.okhttp3:okhttp") {
 version {
 strictly '4.10.0'
 }
}

Существуют либы, которые используют принудительные версии пакетов, это мешает gradle правильно составлять дерево зависимостей.

**Исключение пакета из сборки** и повторная имплементация в корневом build.gradle плагина тоже не поможет:

configurations.configureEach { pkg ->
 pkg.exclude(group: "com.squareup.okhttp3", module: "okhttp")
}