# Настройка рабочего места программиста

### 1. Предварительная подготовка

1. Если у вас Ubuntu 20.04, то делаем следующее:
   1. [устанавливаем дополнительное ПО](020-01-ubuntu-20.04-install-soft.md)
   2. [настраиваем докер](020-02-linux-docker-configure.md)
2. Если у вас Fedora 35+, то делаем следующее:
   1. [устанавливаем дополнительное ПО](../050_prepare_requre_soft/020-03-fedora35-install-soft.md)
   2. [устанавливаем docker](../050_prepare_requre_soft/020-04-fedora35-install-docker.md)
   3. [настраиваем докер](020-02-linux-docker-configure.md)

### 2. Настраиваем доступ к проектам по ключам-SSH

**!!!ВНИМАНИЕ!!!**: тильда(~) - это домашняя директория пользователя

Заходим в папку ~/.ssh/ и смотрим есть ли в ней файлы id\_ed25519, id\_ed25519.pub

Если файлов нет, то запускаем команду:

ssh-keygen -t ed25519

И всё время нажимаем Enter пока не закончится команда - вышеуказанные файлы должны появиться

Далее заходим по адресу: <http://gitlab.greetgo/-/profile/keys>

И в большое поле **Key** вставляем содержимое файла ~/.ssh/id\_ed25519.pub

**!!!ВНИМАНИЕ!!!** убедитесь чтобы у файла было расширение .pub - оно обозначает, что это публичный ключ. Другой файл - приватный ключ - **его показывать НЕЛЬЗЯ (никогда и ни кому)**

Нажимаем кнопку Add Key

### 3. Загружаем проект MyBPM из репозитория и открываем его в Intellij IDEA

Проект находиться по адресу: <http://gitlab.greetgo/mybpm/mybpm.server>

В Intellij IDEA выбираем File -> New -> Project From Version Control...

И указываем URL: [`git@gitlab.greetgo](mailto:%60git@gitlab.greetgo):mybpm/mybpm.server.git`

Запоминаем, где будет находиться проект, и нажимаем волшебную кнопку.

Дальше Intellij IDEA скачает репозиторий и начнёт долго подгружать кучу библиотек.

### 4. Подготовка окружения сервера отладки

Окружение разворачивается с помощью docker-compose, для этого необходимо:

1. в проекте зайти в директорию mybpm\_debug\_env
2. и запустить один из двух скриптов:
3. или скрипт инициации окружения для отладки с пустыми БД, но пригодными для работы

* bash 002-recreate.bash

1. или скрипт инициации окружения для отладки с тестовыми данными

* bash 001-recreate-load-test-data.bash

### 6. Запуск сервера отладки

Для запуска сервера отладки используя Intellij Idea необходимо открыть класс:

kz.greetgo.mybpm.server.debug.launcher.DebugServer

И запустить его как Java Application.

Если же есть необходимость запустить сервер без IDE, то в терминале нужно зайти в директорию проекта и запустить команду:

./gradlew launchDebugServer

После того как появиться сообщение:

2022-01-14T10:54:26.223 k.g.m.s.d.l.DebugServer INFO Started DebugServer in 9.866 seconds   
 JVM running for 10.453

Можно подключаться к серверу - он готов и функционирует.

Остановить сервер можно горячей клавишей: Ctrl+C

# Ubuntu 20.04: установка дополнительного ПО

Команда установит базовые программы необходимые программисты:

sudo apt install openjdk-11-jdk openjdk-11-source git gitk jq docker.io curl

Что здесь установилось:

* openjdk-11-jdk - это основной пакет, где содержится язык java для разработчика
* openjdk-11-source - в этом пакете устанавливаются исходные коды ядра языка java
* git - для работы с репозиториями
* gitk - графическая работа с репозиторием (иногда бывает полезна, чтобы узнать кто-там чего накомитил)
* jq - консольная утилита для обработки данным в формате json
* docker.io - докер
* curl - для rest-операций из bash

Ещё необходимо установить docker-compose для этого надо запустить следующие команды:

sudo curl -L "https://github.com/docker/compose/releases/download/1.29.2/docker-compose-$(uname -s)-$(uname   
-m)" -o /usr/local/bin/docker-compose  
sudo chmod +x /usr/local/bin/docker-compose

# Ubuntu 20.04 : настройка docker

Создаём конфигурационную директорию

sudo mkdir /etc/docker

И настраиваем докер

cat | sudo tee /etc/docker/daemon.json <<EOF  
{  
 "exec-opts": ["native.cgroupdriver=systemd"],  
 "log-driver": "json-file",  
 "log-opts": {  
 "max-size": "100m"  
 },  
"storage-driver": "overlay2",  
"storage-opts": [  
 "overlay2.override\_kernel\_check=true"  
 ]  
}  
EOF

Теперь нужно у докера отрубить memlock, для этого открываем файл:

sudo mcedit -b /usr/lib/systemd/system/docker.service

Если нет файла: /usr/lib/systemd/system/docker.service То его место можно найти по команде:

systemctl status docker

В открытом файле находим строку:

ExecStart=/usr/bin/dockerd -H fd:// --containerd=/run/containerd/containerd.sock

И добавляем в её конец строку:

--default-ulimit memlock=-1:-1

Чтобы получилась строка:

ExecStart=/usr/bin/dockerd -H fd:// --containerd=/run/containerd/containerd.sock --default-ulimit memlock=-1:-1

Перезапускаем докер:

sudo systemctl daemon-reload  
sudo systemctl restart docker

Также, чтобы не использовать sudo для команды docker, надо добавить пользователя в группу docker

sudo usermod -aG docker $USER

Проверить отключение лимитов на память можно командой:

docker run --rm busybox:1.28 cat /proc/1/limits

Должна появиться строка

...  
Max locked memory unlimited unlimited bytes   
...

Эти ограничения должны быть unlimited (т.е. ограничений нет)

# Альтернативные запуски DebugServer-а

## [Запуск с внешним ElasticSearch](002-run-debug-server-with-extern-es.md)

# Запуск DebugServer-а с внешним ElasticSearch

* Запустить команду:

cd КОРЕНЬ\_ПРОЕКТА/mybpm\_debug\_env  
  
bash 002-recreate.bash

Если эта команда уже ранее запускалась, то запускать её не нужно

* Запустить команду:

cd КОРЕНЬ\_ПРОЕКТА/mybpm\_debug\_env  
  
bash remove.bash

* 1. Зайти в директорию:

cd ~/.local/share/mybpm/app\_skeleton

* 2. Создать в ней папку:

mkdir conf

* 3. Скопировать в папку файл

cp ../app/conf/ElasticSearchConfig.conf conf/

* 4. Настроить этот файл (~/.local/share/mybpm/app\_skeleton/conf/ElasticSearchConfig.conf)

Файл выглядит примерно так:

# Этот конфиг нужен для того, чтобы перенаправлять среду разработки в другое место, чем изначальное.  
# Например, это может оказаться полезным при направлении на кластер MongoDB развёрнутый где-то в другом месте.  
#   
# Параметры доступа определены:   
# 1) для тестового режима;   
# 2) для пересоздания БД (recreate);   
# 3) для Debug-сервера   
  
# Доступы к серверам elastic  
#servers=  
# Пользователь, или пусто, если аутентификация не нужна  
#user=  
# Пароль, если указан пользователь  
#pass=  
# Путь и имя файла ca.crt  
#caCrtFilePath=

Нужно настроить параметры доступа к эластику. Ниже представлен пример как это можно сделать. Обратите внимание, что символы решётки у этих параметров убраны.

# Этот конфиг нужен для того, чтобы перенаправлять среду разработки в другое место, чем изначальное.  
# Например, это может оказаться полезным при направлении на кластер MongoDB развёрнутый где-то в другом месте.  
#   
# Параметры доступа определены:   
# 1) для тестового режима;   
# 2) для пересоздания БД (recreate);   
# 3) для Debug-сервера   
  
# Доступы к серверам elastic  
servers=https://192.168.17.101:30301,192.168.17.102:30302,192.168.17.103:30303  
# Пользователь, или пусто, если аутентификация не нужна  
user=es\_user  
# Пароль, если указан пользователь  
pass=sUpEг-$eKгe+  
# Путь и имя файла ca.crt  
caCrtFilePath=/root/ca/ca.crt

1. Запустить файл ещё раз

cd КОРЕНЬ\_ПРОЕКТА/mybpm\_debug\_env  
  
bash 002-recreate.bash

Данный скрипт уже будет обращаться к внешнему ElasticSearch.

После этого можно запустить DebugServer - и он уже будет работать с внешней БД.

Чтобы вернуть как было: удалите папку app\_skeleton и запустите 002-recreate.bash.

# 1) Установка MyBPM на кластере Kubernetes.

Перед установкой необходимо получить доступы к следующим сервисам:

* Apache Zookeeper
* Apache Kafka
* MongoDB
* ElasticSearch
* PostgreSQL
* Кластер Kubernetes

## 1.1) Образы платформы

Предварительно нужно получить актуальные на данный момент версии платформы - это имена образов docker. На данный момент актуальными являются следующие образы:

* out.mybpm.kz/mybpm-api-dev:4.5.1215 - API-сервис платформы MyBPM
* out.mybpm.kz/mybpm-web-dev:4.5.1111 - Web-представление платформы MyBPM

В вашем случае имена образов могут поменяться. Не забудьте в дальнейшем заменять эти названия на ваши.

## 1.2) Подготовка пространства имён

В этом руководстве используется пространство имён:

mybpm-platform-test

У вас оно должно быть другим и отражать суть кластера, например:

mybpm-prod-stone

Которое может обозначать платформу MyBPM на рабочей среде в проекте Stone - выберите понятное вам название. В дальнейшем в этом руководстве везде заменяйте mybpm-platform-test на ваше.

Создать можно это пространство имён выполнив команду:

kubectl apply -f kube-yaml-files/010-prepare/10-ns.yaml

## 1.3) Подготовка секретного ключа для загрузки образа из репозитория

Образы платформы MyBPM загружаются из репозитория dockerhub.mybpm.kz и доступ к нему должен быть предоставлен. Вам предоставят файл, с ключом доступа. Пример такого файла находиться здесь:

kube-yaml-files/020-MyBPM-in-kube/05-auth-out-mybpm-kz.json

В файле:

kube-yaml-files/020-MyBPM-in-kube/06-create-docker-secret.bash

Находиться скрипт, чтобы его применить. В нём указано пространство имен: mybpm-platform-test - поменяйте его на ваше и потом запустите его:

bash kube-yaml-files/020-MyBPM-in-kube/06-create-docker-secret.bash

Данный скрипт создаст в Kubernetes объект

secret/out-mybpm-kz

Через который Kubernetes будет обращаться к репозиторию out.mybpm.kz для скачивания образов. Этот kube-объект в дальнейшем будет указываться в yaml-файлах.

## 1.4) Распределение по хостам кластера Kubernetes

Для того чтобы контролировать, на каких хостах кластера Kubernetes будут запускаться сервисы платформы MyBPM нужно создать соответствующие метки на хостах, и потом запускаемые контейнеры привязать к этим меткам.

Для настройки соответствующих меток имеется скрипт:

bash kube-yaml-files/020-MyBPM-in-kube/07-distribute-node-labels.bash

В нём есть текст:

setup\_node master1 +api -web  
setup\_node master2 +api -web  
setup\_node master3 -api -web  
setup\_node worker4 -api -web  
setup\_node worker5 -api +web  
setup\_node worker6 -api +web

Который добавляет метку api=ok на хосты master1 и master2, а метку web=ok на хосты: worker5 и worker6. В вашем случае может быть по другому - откорректируйте данный кусок скрипта, как вам нужно. Теперь его нужно запустить.

Проверить распределение меток можно с помощью команды:

kubectl get node --show-labels

## 1.5) API-сервис платформы MyBPM

Теперь нужно развернуть API-сервис платформы MyBPM

# Развёртка платформы MyBPM на кластере kubernetes

## 7) Подготовка к развёртыванию

Прежде чем устанавливать платформу MyBPM необходимо подготовить для неё хранилища дынных, а именно:

* 7.0.1) Apache Zookeeper - [Развёртывание кластера Zookeeper]
* 7.0.2) Apache Kafka - [Развёртывание кластера Kafka]
* 7.0.3) MongoDB - [Развёртывание кластера MongoDB]
* 7.0.4) Elastic Search - [Развёртывание кластера Elastic Search]
* 7.0.5) PostgreSQL - [Развёртывание кластера PostgreSQL]
* 7.0.6) Кластер Kubernetes - [Установка kubernetes с тремя мастерами и "плавающим" IP-мастера]

А также развернуть кластер Kubernetes:

### 7.1) Настройка kubectl локальном компьютере

Разворачивание платформы можно делать на локальном компьютере, который имеет доступ на кластер по порту 6443. Для этого на компьютере в домашней директории необходимо создать директорию .kube и в ней создать файл config. Содержимое этого файла нужно взять из файла /etc/kubernetes/admin.conf с мастера, например первого.

После того как появиться этот файл на вашем компьютере заработает команда:

kubectl get pod --all-namespaces -o wide

### 7.2) Подготовка директории yaml-файлов

Управление кластером kubernetes осуществляется с помощью yaml-файлов. В проекте mybpm-doc имеется директория:

kube-yaml-files/010-several-host-servers

Её необходимо скопировать к себе на локальный компьютер, например в директорию yaml-files - далее она так и будет назваться.

### 7.2) Настройка пространства имён

Внутри kubernetes программы разделены на пространства имён, и каждая система имеет своё пространство имён, в том числе и платформа MyBPM тоже должна иметь своё пространство имён. Для него нужно придумать наименование, например:

mybpm-platform-test

В файле:

yaml-files/010-prepare/10-ns.yaml

Переименуйте пространство имён на то, которое используется у вас

Далее это пространство имён нужно создать, применив полученный файл в kubernetes с помощью команды:

kubectl apply -f yaml-files/010-prepare/10-ns.yaml

Чтобы посмотреть, сработала ли команда, можно командой:

kubectl get ns

В списке должна вылететь ваше пространство имён:

NAME STATUS AGE  
default Active 3h28m  
kube-node-lease Active 3h28m  
kube-public Active 3h28m  
kube-system Active 3h28m  
mybpm-platform-test Active 5s