**МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ**

**федеральное государственное автономное образовательное учреждение высшего образования**

**Санкт-Петербургский национальный исследовательский университет информационных технологий, механики и оптики**

**Мегафакультет трансляционных информационных технологий**

**Факультет информационных технологий и программирования**

Отчет

По дисциплине практике в компании «Софтверке»

Создание консольного приложения на OSGi

Выполнил студент группы №M33122  
*Чикалёв Никита Александрович*

*САНКТ-ПЕТЕРБУРГ*

*2022*

## **Цель задания**

Познакомить студента с базовыми концепциями OSGi, попробовать эти концепции на примерах и с их использованием создать небольшое консольное приложение (этап 5).

## **Описание**

Практика состоит из 5 этапов. На первом этапе требуется познакомиться с OSGi, Apache Felix. На 2-4 этапах требуется создать несколько простых приложений, чтобы на практике познакомиться с основными концепциями. На 5-м этапе требуется создать небольшое, но полноценное приложение, применив знания и умения, полученные на этапах 1-4.

## **Этапы выполнения задания**

### *Этап 1. Подготовительный*

1. Прочитайте главы 1-4 «OSGi in Action». Постарайтесь понять основные концепции OSGi: modularity, lifecycle, bundles, services, components и т.п.
2. Скачайте и установите [Apache Felix](http://felix.apache.org/downloads.cgi). Установите Felix Web Console.
3. Запустите Apache Felix в режиме debug:  
   java -Xdebug -Xrunjdwp:transport=dt\_socket,server=y,suspend=n,address=<port-number> -jar bin/felix.jar

### *Этап 2. Реализация OSGi-сервиса*

1. Создайте бандл, содержащий сервис (интерфейс с одним методом и его реализацию). Зарегистрируйте сервис, используя активатор бандла. Метод должен выводить на консоль сообщение «Hello OSGi World!».
2. Установите бандл в Felix.
3. Создайте новый бандл. Бандл должен потреблять созданный ранее сервис. Метод сервиса, созданного на 1 шаге, должен вызываться при активации нового бандла.
4. Установите бандл в Felix. Убедитесь, что метод сервиса был вызван.

### *Этап 3. Apache Felix Service Component Runtime*

Аналогично этапу 2, создайте два бандла: который содержит реализацию сервиса и который использует этот сервис. Используйте [Apache Felix Service Component Runtime](http://felix.apache.org/documentation/subprojects/apache-felix-service-component-runtime.html), [SCR Annotations](http://felix.apache.org/documentation/subprojects/apache-felix-maven-scr-plugin/scr-annotations.html) и [Apache Felix Maven SCR Plugin](https://felix.apache.org/documentation/subprojects/apache-felix-maven-scr-plugin/apache-felix-maven-scr-plugin-use.html) для сборки.

### *Этап 4. Создание собственной команды для Apache Felix Gogo*

Требуется создать собственную команду «practice:hello» с одним параметром, которая при вызове печатает на консоль «Hello, <param>», где <param> - введенный пользователем параметр.

См. <http://coders-kitchen.com/2012/04/06/tutorial-an-own-gogo-shell-command/>

### *Этап 5. Создание приложения*

Требуется создать приложение, которое получает через API новостных порталов список актуальных новостей и выводит на консоль 10 самых часто встречающихся слов из заголовков новостей. Данные берутся из следующих источников:

1. Lenta.ru API (<https://api.lenta.ru/rss>).
2. RSS АиФ (<http://www.aif.ru/rss/news.php>).
3. Прочие источники на усмотрение студента.

Процесс подсчета инициируется пользователем приложения с помощью консольной команды «news:stats». Пользователь может передать источник в качестве параметра. Если команда вводится без параметров, пользователю предлагается выбрать источник данных (один из доступных в системе, или все сразу). Список источников выводится на консоль.

Если пользователь выбирает незарегистрированный (недоступный) источник данных, в системе не зарегистрировано ни одного источника или сеть недоступна, пользователю выводится соответствующее сообщение.

Реализация получения данных из каждого источника должна быть отдельной имплементацией общего для всех источников сервиса (интерфейса).

**Этап 1. Подготовительный**

1. Были успешно прочитаны главы 1-4 «OSGi in Action». Данный пункт помог разобраться в том, что такое OSGi, а также он положил основы для выполнения последующих этапов, поскольку в книге были вполне понятные примеры кода, как и сама теоретическая часть.
2. Был скачан и запущен [Apache Felix](http://felix.apache.org/downloads.cgi).

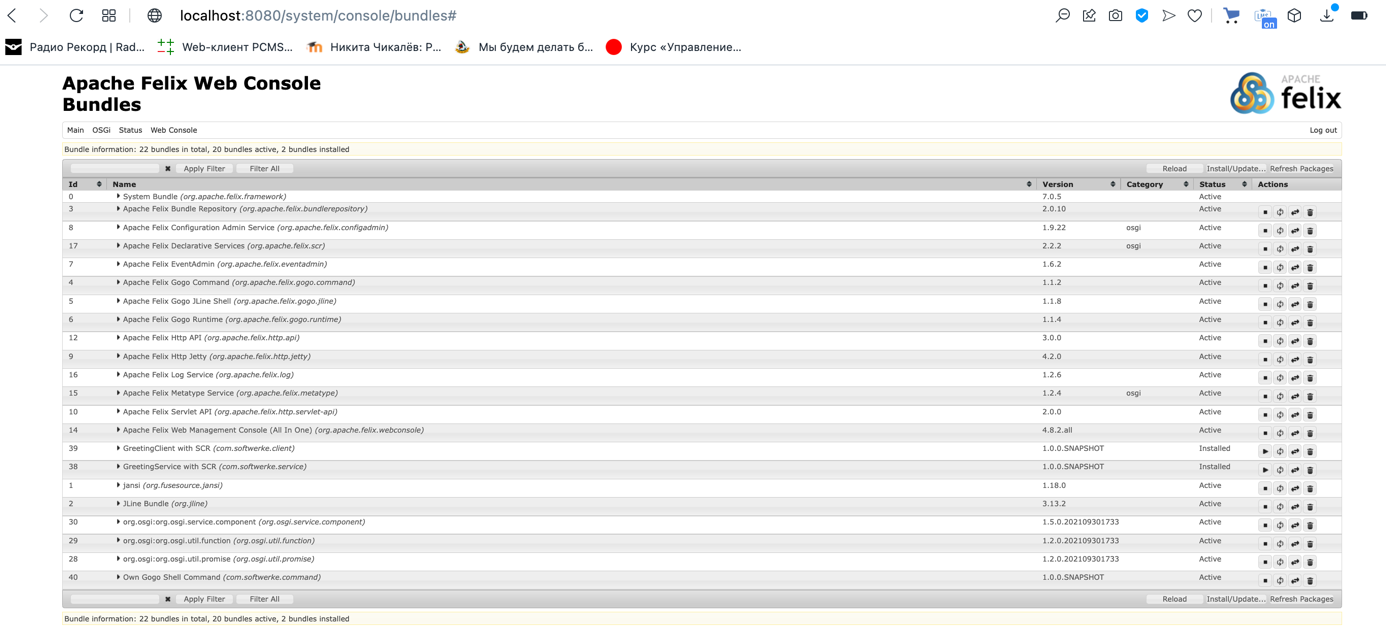
Изображение выглядит как стол

Автоматически созданное описание

Также была успешно установлена Felix Web Console.

Изображение выглядит как текст

Автоматически созданное описание



*Вывод по этапу 1:*

Данный этап помог познакомиться с OSGi, с понятием «бандл» и [Apache Felix](http://felix.apache.org/downloads.cgi), а также подготовить место для работы с будущими бандлами, которые мы будем создавать на последующих этапах.

### **Этап 2. Реализация OSGi-сервиса**

1. Создали первый бандл с интерфейсом сервиса, имплементацией сервиса и активатором данного сервиса.Изображение выглядит как текст

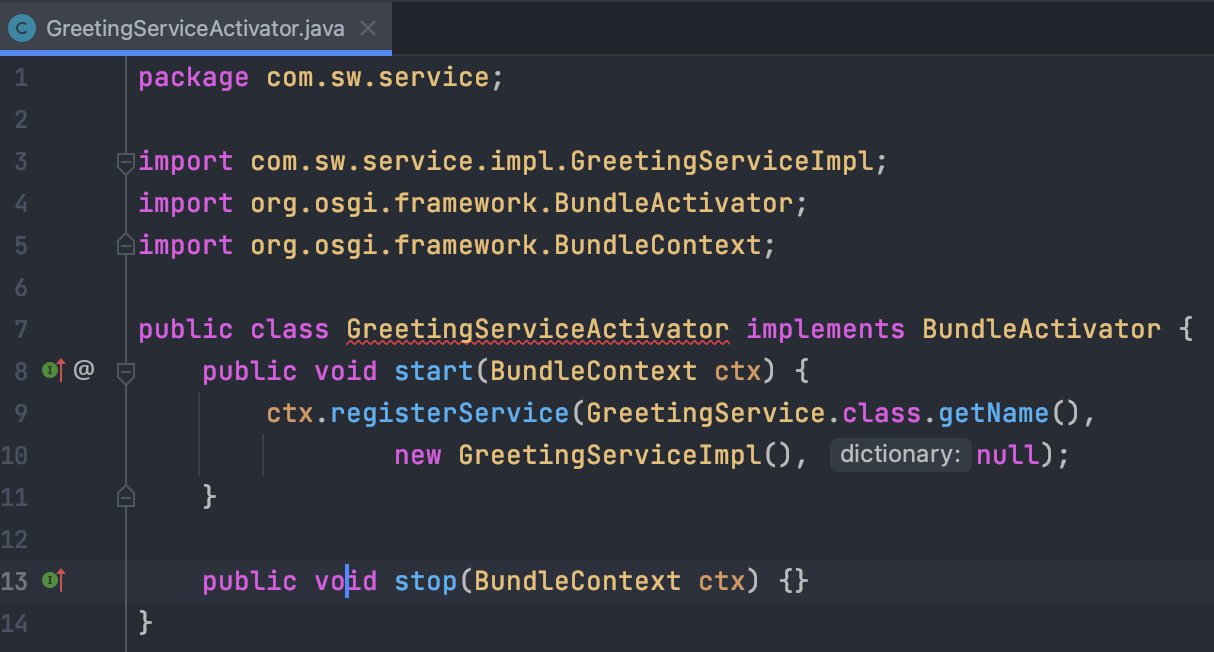
   Автоматически созданное описание

Изображение выглядит как текст

Автоматически созданное описание

Изображение выглядит как текст

Автоматически созданное описание





1. Установили этот бандл в Felix.

Изображение выглядит как текст

Автоматически созданное описание

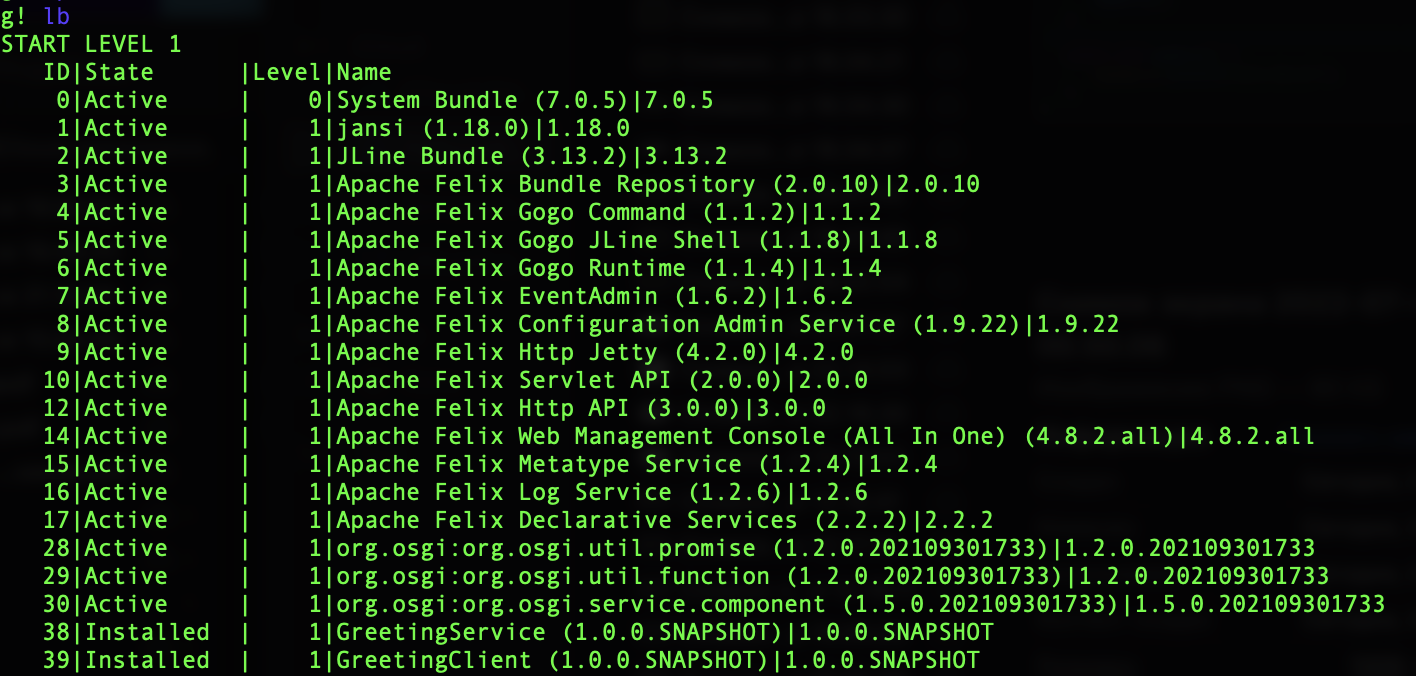
1. Создали новый бандл, который является клиентом для нашего сервиса. При активации он вызывает метод нашего созданного ранее сервиса.

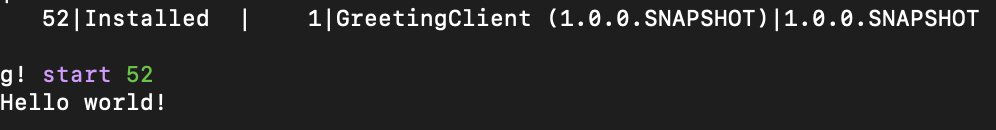
Изображение выглядит как текст

Автоматически созданное описание



1. Установили наш клиент в Felix и убедились, что он работает.





*Вывод по этапу 2:*

На данном этапе, мы на практике научились создавать, устанавливать и запускать свои собственные бандлы с простой реализацией. Подкрепили понятия “BundleActivator”, “Service”, “BundleContext”.

### **Этап 3. Apache Felix Service Component Runtime**

Как и в этапе 2 мы создали два бандла, однако в этот раз мы использовали [Apache Felix Service Component Runtime](http://felix.apache.org/documentation/subprojects/apache-felix-service-component-runtime.html), [SCR Annotations](http://felix.apache.org/documentation/subprojects/apache-felix-maven-scr-plugin/scr-annotations.html) и [Apache Felix Maven SCR Plugin](https://felix.apache.org/documentation/subprojects/apache-felix-maven-scr-plugin/apache-felix-maven-scr-plugin-use.html) для сборки.

Изображение выглядит как текст

Автоматически созданное описание

Изображение выглядит как текст

Автоматически созданное описание

Как мы видим из скриншота выше, нам уже не понадобился класс активатора, также нам уже не нужно регистрировать сервис руками, вместо этого все за нас делают декораторы.

Изображение выглядит как текст

Автоматически созданное описание

Далее мы установили их и проверили работоспособность данных бандлов.

Изображение выглядит как текст

Автоматически созданное описание

*Вывод по этапу 3:*

*Данный этап познакомил нас с аннотациями, мы теоретически и практически смогли воспользоваться их функциями. Они действительно очень упрощают разработку, поскольку в разы уменьшают написание кода, а также реализуют дополнительный функционал.*

### **Этап 4. Создание собственной команды для Apache Felix Gogo**

На данном этапе мы создавали банл, реализующий свою собственную Gogo Shell команду.

Реализация самого бандла:

Изображение выглядит как текст

Автоматически созданное описание

Здесь нам не понадобилось создавать клиентов и интерфейсы, здесь мы смогли обойтись одним лишь сервисом.

Установим и проверим функионал:

Изображение выглядит как текст

Автоматически созданное описание

*Вывод по этапу 4:*

*На данном этапе мы познакомились с созданием бандла, который реализует нашу собственную команду Gogo Shell. Данный этап оказался очень полезен, поскольку с его помощью нам стало понятно, как реализовывать взаимодействие с пользователем в ключе использования функционала наших бандлов.*

**Вывод:**

В процессе прохождения данной практики нам удалось поработать с совершенно новыми для нас технологиями на всех этапах. Мы на деле узнали, как происходит работа с новыми знаниями, поскольку пришлось учить как сам OSGi, так и Maven. К сожалению, не удалось выполнить этап 5, из-за недостатка навыков и времени, но все равно было очень увлекательно и познавательно. В процессе прохождения мы познакомились с OSGi, Maven, Gogo Shell, Apache Felix, как и со всей системой модульности в целом, а также бандлами.