

Разработчик Java

Перед вами первый курс OTUS. Он стартовал в апреле 2017 года. С тех пор идет с неизменным успехом и на настоящий момент насчитывает более 10 запусков.

Начало занятий

В июне

Длительность курса: 148 академических часов

1 Язык и платформа Java

1 Подготовка к курсу. ДЗ

Познакомиться с программой курса.
Изучить основные инструменты.

Домашние задания

1 Проект maven с модульной структурой

- 1) Создать аккаунт на github.com (если еще нет)
- 2) Создать репозиторий для домашних работ
- 3) Сделать checkout репозитория на свой компьютер
- 4) Создайте локальный бранч hw01-maven
- 5) Создать проект maven
- 6) В проект добавьте последнюю версию зависимости
`<groupId>com.google.guava</groupId>`
`<artifactId>guava</artifactId>`
- 7) Создайте модуль hw01-maven
- 8) В модуле сделайте класс HelloOtus
- 9) В этом классе сделайте вызов какого-нибудь метода из guava
- 10) Добавьте нужный плагин maven и соберите "толстый-jar"
- 11) Убедитесь, что "толстый-jar" запускается.
- 12) Сделайте pull-request в gitHub
- 13) Ссылку на PR отправьте на проверку.

2 Дополнение к maven, история изменения языка

Познакомиться со Shade Plugin
Углубить знания о maven
Познакомиться с текущей ситуацией в мире java

3 Контейнеры и алгоритмы. ДЗ

Познакомиться с Generic-ами в Java
Познакомиться со стандартными коллекциями.

Домашние задания

1 DIY ArrayList

Написать свою реализацию ArrayList на основе массива.

```
class DIYArrayList<T> implements List<T>{...}
```

Проверить, что на ней работают методы из java.util.Collections:

```
Collections.addAll(Collection<? super T> c, T... elements)
```

```
Collections.static <T> void copy(List<? super T> dest, List<? extends T> src)
```

```
Collections.static <T> void sort(List<T> list, Comparator<? super T> c)
```

1) Проверьте на коллекциях с 20 и больше элементами.

2) DIYArrayList должен имплементировать ТОЛЬКО ОДИН интерфейс - List.

3) Если метод не имплементирован, то он должен выбрасывать исключение UnsupportedOperationException.

4 Инструменты для преобразования контейнеров, unsafe, jmh

5 QA и тестирование

Домашние задания

1 Свой тестовый фреймворк.

Написать свой тестовый фреймворк.

Поддерживать свои аннотации @Test, @Before, @After.

Запускать вызовом статического метода с именем класса с тестами.

Т.е. надо сделать:

1) создать три аннотации - @Test, @Before, @After.

2) Создать класс-тест, в котором будут методы, отмеченные аннотациями.

3) Создать "запускалку теста". На вход она должна получать имя класса с тестами.

4) "Запускалка" должна в классе-тесте найти и запустить методы, отмеченные аннотациями.

5) Алгоритм запуска должен быть такой:

метод Before

методы Test

метод After

для каждой такой "тройки" надо создать СВОЙ объект класса-теста.

6) Исключение в одном тесте не должно прерывать весь процесс тестирования.

7 **Углубленные основы (примитивные типы, Remote debug, Hot swap).**

Домашние задания

1 Автомагическое логирование.

Разработайте такой функционал: метод класса можно пометить самодельной аннотацией @Log, например, так:

```
class TestLogging {  
    @Log  
    public void calculation(int param) {}  
}
```

При вызове этого метода "автомагически" в консоль должны логироваться значения параметров.

Например так.

```
class Demo {  
    public void action() {  
        new TestLogging().calculation(6);  
    }  
}
```

В консоле должно быть:

executed method: calculation, param: 6

Обратите внимание: явного вызова логирования быть не должно.

1 Сравнение разных сборщиков мусора

Написать приложение, которое следит за сборками мусора и пишет в лог количество сборок каждого типа (young, old) и время которое ушло на сборки в минуту.

Добиться OutOfMemory в этом приложении через медленное подтекание по памяти (например добавлять элементы в List и удалять только половину).

Настроить приложение (можно добавлять Thread.sleep(...)) так чтобы оно падало с ООМ примерно через 5 минут после начала работы.

Собрать статистику (количество сборок, время на сборки) по разным GC.

!!! Сделать выводы !!!
ЭТО САМАЯ ВАЖНАЯ ЧАСТЬ РАБОТЫ:
Какой gc лучше и почему?

1 Концепты проектирования ООП. ДЗ

Домашние задания

1 Эмулятор банкомата

Написать эмулятор АТМ (банкомата).

Объект класса АТМ должен уметь:

- принимать банкноты разных номиналов (на каждый номинал должна быть своя ячейка)
- выдавать запрошенную сумму минимальным количеством банкнот или ошибку если сумму нельзя выдать

Это задание не на алгоритмы, а на проектирование.

Поэтому оптимизировать выдачу не надо.

- выдавать сумму остатка денежных средств
-

2 Behavioral patterns

3 **Structural patterns. ДЗ**

Домашние задания

1 Департамент АТМ

Написать приложение АТМ Департамент:

1) Департамент может содержать несколько АТМ.

2) Департамент может собирать сумму остатков со всех АТМ.

3) Департамент может инициировать событие – восстановить состояние всех АТМ до начального (начальные состояния у разных АТМ могут быть разными).

Это тренировочное задание на применение паттернов.

Попробуйте использовать как можно больше.

4 **Creational patterns**

1 Сериализация. ДЗ

Домашние задания

1 Свой json object writer

Напишите свой json object writer (object to JSON string) аналогичный gson на основе javax.json.

Поддержите:

- массивы объектов и примитивных типов
 - коллекции из стандартной библиотеки.
-

2 NIO.XML

3 JDBC. ДЗ

Домашние задания

1 Самодельный ORM

Работа должна использовать базу данных H2. Создайте в базе таблицу User с полями:

- id bigint(20) NOT NULL auto_increment
- name varchar(255)
- age int(3)

Создайте свою аннотацию @Id

Создайте класс User (с полями, которые соответствуют таблице, поле id отметьте аннотацией).

Напишите JdbcTemplate, который умеет работать с классами, в которых есть поле с аннотацией @Id.

Executor должен сохранять объект в базу и читать объект из базы.

Имя таблицы должно соответствовать имени класса, а поля класса - это колонки в таблице.

Методы JdbcTemplate'a:

```
void create(T objectData);
```

```
void update(T objectData);
```

```
void createOrUpdate(T objectData); // опционально.
```

```
<T> T load(long id, Class<T> clazz);
```

Проверьте его работу на классе User.

Метод createOrUpdate - необязательный.

Он должен "проверять" наличие объекта в таблице и создавать новый или обновлять.

Создайте еще одну таблицу Account:

- no bigint(20) NOT NULL auto_increment
- type varchar(255)
- rest number

Создайте для этой таблицы класс Account и проверьте работу JdbcTemplate на этом классе.

4 Общие вопросы работы с СУБД, myBatis

5 Hibernate. ДЗ

Домашние задания

1 Использование Hibernate

Работа должна использовать базу данных H2.

Возьмите за основу предыдущее ДЗ

(Самодельный ORM)

и реализуйте функционал сохранения и чтения объекта User через Hibernate.

(Рефлексия больше не нужна)

Конфигурация Hibernate должна быть вынесена в файл.

Добавьте в User поля:

адрес (OneToOne)

```
class AddressDataSet {  
    private String street;  
}
```

и телефон (OneToMany)

```
class PhoneDataSet {  
    private String number;  
}
```

Разметьте классы таким образом, чтобы при сохранении/чтении объекта User каскадно сохранялись/читались вложенные объекты. Не забывайте про сохранение абстракций в приложении (см. комментарий в вебинаре).

6 JPQL

7 **Типы ссылок.
Кэширование.
ДЗ**

Домашние задания

1 Свой cache engine

Напишите свой cache engine с soft references.
Добавьте кэширование в DBService из задания
про Hibernate ORM

8 **No SQL**

9 **Web сервер.
ДЗ**

Домашние задания

1 Веб сервер

Встроить веб сервер в приложение из ДЗ про
Hibernate ORM.
Сделать админскую страницу, на которой админ
должен авторизоваться.
На странице должны быть доступны следующие
функции:
- создать пользователя
- получить список пользователей

10 **Asynchronous
Web
applications**

11 **Dependency injection. ДЗ**

Домашние задания

1 Приложение с IoC контейнером

Собрать war для приложения из предыдущего ДЗ.

Создавать кэш и DBService как Spring beans, передавать (inject) их в сервлеты.

Запустить веб приложение во внешнем веб сервере.

12 **Логгирование**

1 14.1 Thread

Многопоточность.
Класс и объект Thread.
Создание многопоточного приложения.
Проблемы многопоточного доступа.

2 14.2 JMM. ДЗ

Основная проблема многопоточности
«Железные» оптимизации
«Программные» оптимизации
JMM
Volatile
Happens before
Lock-free алгоритмы, CAS

Домашние задания

1 Последовательность чисел

Два потока печатают числа от 1 до 10, потом от 10 до 1.

Надо сделать так, чтобы числа чередовались, т.е. получился такой вывод:

Поток 1: 1 2 3 4 5 6 7 8 9 10 9 8 7 6 5 4 3 2 1 2 3
4....

Поток 2: 1 2 3 4 5 6 7 8 9 10 9 8 7 6 5 4 3 2 1 2
3....

3 **Message System.** **ДЗ**

Домашние задания

1 MessageSystem

Добавить систему обмена сообщениями в ДЗ
про веб сервер с IoC контейнером.

Пересылать сообщения из вебсокета в
DBService и обратно.

Организовать структуру пакетов без
циклических зависимостей.

4 **Executors**

5 Многопроцессные приложения. ДЗ

Домашние задания

1 MessageServer

Сервер из предыдущего ДЗ про MessageSystem разделить на три приложения:

- MessageServer
- Frontend
- DBServer

Запускать Frontend и DBServer из MessageServer.

Сделать MessageServer сокет-сервером, Frontend и DBServer клиентами.

Пересылать сообщения с Frontend на DBService через MessageServer.

Запустить приложение с двумя серверами фронтенд и двумя серверами баз данных на разных портах.

Если у вас запуск веб приложения в контейнере, то MessageServer может копировать root.war в контейнеры при старте

6 NIO

1 Консультация по ДЗ и проектам

Домашние задания

1 Проектная работа

Заключительный месяц курса посвящен проектной работе. Свой проект это то, что интересно писать студенту. То, что можно создать на основе знаний, полученных на курсе. При этом не обязательно закончить его за месяц. В процессе написания по проекту можно получить консультации преподавателей.

Проект должен стать примером кода, который можно показывать потенциальным работодателям.

Примеры тем проекта:

- web сервер (разберите протокол)
 - socket сервер на NIO (как netty)
 - свой ORM
 - распределенный кэш
 - кэш для hibernate
-

2 Консультация по ДЗ и проектам

3 Защита проектов