OTUS

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.

Длительность курса: 150 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 и 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) Проверяйте 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**	На примере изучить принципы создания коллекций. Познакомиться с пакетом unsafe, утилитой JMH и популярными библиотеками коллекций.
5	**QA и тестирование**	Познакомиться с junit и mockito На примере понять, что такое "тестируемое приложение"

6 **Аннотации. ДЗ**

Познакомиться с механизмом Reflection.
Узнать что такое Аннотации и как их можно сделать

Домашние задания

1 Свой тестовый фреймворк.

Написать свой тестовый фреймворк.

Поддержать 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, в котором следует найти и запустить методы отмеченные аннотациями и пункта 1.
4) Алгоритм запуска должен быть следующий::
метод(ы) Before
текущий метод Test
метод(ы) After
для каждой такой "тройки" надо создать СВОЙ объект класса-теста.
5) Исключение в одном тесте не должно прерывать весь процесс тестирования.
6) На основании возникших во время тестирования исключений вывести статистику выполнения тестов (сколько прошло успешно, сколько упало, сколько было всего)

7	**Углубленные основы (примитивные типы, Remote debug, Hot swap).**	Узнать детали устройства типов данных в Java. Познакомиться с механизмами Remote Debug и Hot swap. Знакомство с утилитой Jol

8	**Байт код, class-loader, инструменатция, asm. ДЗ**	Познакомиться с принципами работы виртуальной машины Java, ClassLoader-ами и байт-кодом
		
		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 дожно быть:
		executed method: calculation, param: 6
		
		Обратите внимание: явного вызова логирования быть не должно.

9 **Сборщик мусора. ДЗ**

Знакомство со сборщиком мусора в Java

Домашние задания

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 OOM 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?

Выводы оформить в файле Conclusions.md в корне папки проекта.
Результаты измерений сведите в таблицу.

---

10 **Java 8**

Введение в функциональное программироавние (ФП).
Знакомство с возможностями ФП, которые появились в Java 8.

2 Проектирование

1 **Концепты проектирования ООП. ДЗ**

Изучить принципы SOLID и общие критерии идеальной архитектуры

Домашние задания

1 Эмулятор банкомата

Написать эмулятор ATM (банкомата).

Объект класса ATM 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**

Изучить поведенческие паттерны проектирования

---

3 **Structural patterns. ДЗ**

Изучить структурные паттерны проектирования

Домашние задания

1 Департамент ATM

Написать приложение ATM Департамент:
1) Департамент может содержать несколько ATM.
2) Департамент может собирать сумму остатков со всех ATM.
3) Департамент может инициировать событие – восстановить состояние всех
ATM до начального (начальные состояния у разных ATM могут быть
разными).
Это тренировочное задание на применение паттернов.
Попробуйте использовать как можно больше.

---

4 **Creational patterns**

Изучить "создающие" паттерны проектирования

3 Работа с окружением

1 **Сериализация. ДЗ**

Познакомиться с функционалом сериализации объектов

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 стандартный библиотеки.

---

2 **NIO. Логирование**

познакомиться с методами логирования в Java.
познакомиться с NIO

---

3 **JDBC. ДЗ**

Познакомиться с транзакцией в реляционной СУБД и jdbc

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 котрых 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**

Рассмотреть CAP-теорему
Рассмотреть методы организации блокировок
Познакомиться с MyBatis

---

5 **Hibernate. ДЗ**

Закомство с Hibernate

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.
(Рефлейсия 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 объека User каскадно сохранялись/читались вложенные объекты.
Не забывайте про сохранение абстракций в приложении (см. комментарий в вебинаре).

---

6 **JPQL**

Познакомиться с Connection Pool
Узнать методы конструирования запросов в Hibernate

7	**Типы ссылок. Кэширование. ДЗ**	Узнать какие в java есть виды ссылок и для чего они нужны Понять как устроены кэши Познакомиться с "промышленным" кэшом Ehcache 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**	Познакомиться с noSQL базами данных Понять отличия SQL от noSQL, когда и что следует использовать. Познакомится с MongoDB
9	**Web сервер. ДЗ**	На примере Jetty понять принципы работы Web-сервера и servlet API 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 P.S. при желании вместо Hibernate и H2 можно использовать MongoDB

10	**Dependency injection. ДЗ**	Изучить принципы работы контейнера TomCat Изучить принципы работы framework Spring 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.
11	**Asynchronous Web applications**	Узнать как можно сделать ассинхронный web-сервис на java. Познакомиться со Spring Boot

4 Многопоточность

1 **Thread**

Познакомиться с основными принципами многопоточности
Узнать как управлять потоками в Java

---

2 **JMM. ДЗ**

Познакомиться с основными проблемами многопоточности.
Понять зачем придумали JMM
Узнать основные положения JMM

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**	Познакомиться с потокобезопасными контейнерами Познакомиться с паттерном - "система обмена сообщениями" 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**	Познакомиться с пулами потоков в Java

5	**Многопроцессные приложения. ДЗ**	Изучение сетевого взаимодействия в java. Изучение принципов работы "клиент-серверного" приложения в Java 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**	Изучение основ сетевых возможностей NIO
7	**Netty**	изучить основные принципы работы Netty.

5 Проектная работа

1	**Консультация по ДЗ и проектам**	получить ответы на вопросы по проекту 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**	получить ответы на вопросы по проектной работе
3	**Защита проектов**	защитить свой проект и получить рекомендации экспертов