

Практика по Java

Введение

Юрий Литвинов
yurii.litvinov@gmail.com

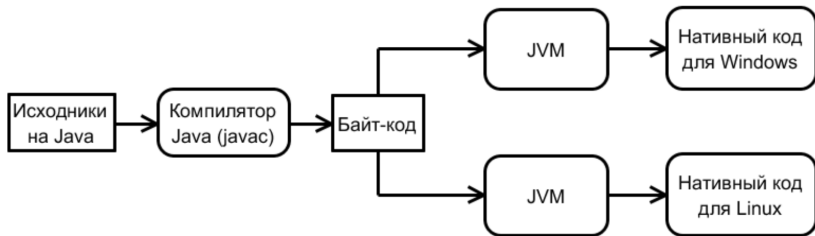
16.01.2019г

Формальности

- ▶ Чтобы получить зачёт, надо:
 - ▶ Сдать некоторый (большой!) процент домашних работ
 - ▶ Успешно написать две контрольные
 - ▶ Решать задачи прямо на паре
 - ▶ По каждой домашке дедлайн порядка двух недель
 - ▶ Решения оцениваются от 0 до 10 баллов
- ▶ Условия, материалы и сдача домашних через <http://hwproj.me/>
- ▶ Среда программирования — какая угодно
 - ▶ Рекомендуется IntelliJ IDEA
- ▶ Между “сделать” и “сдать” большая разница

Язык Java

- ▶ Появился в 1995 году, актуальная версия — Java 11
- ▶ Объектно-ориентированный язык с сильной типизацией
- ▶ Прежде всего — для разработки прикладного ПО (в отличие от C++)
- ▶ Использует виртуальную машину (compile once — run everywhere, опять же в отличие от C++)
- ▶ Just-in-time-компиляция



Особенности

- ▶ Сборка мусора
 - ▶ Это не значит, что за памятью можно не следить!
- ▶ Практически всё — объект
- ▶ Стандартизация элементарных типов (в отличие от C++)
- ▶ Пакеты и библиотеки (имена пакетов стандартизованы, например, `com.example.myclasses`)
- ▶ Рефлексия
- ▶ Некоторая поддержка функционального стиля
- ▶ Несколько странная реализация шаблонов (генерики)

Mandatory slide про стандартные числовые типы

Тип	Значения	Размер
<i>byte</i>	$-2^7..2^7 - 1$ ($-128..127$)	8 бит
<i>short</i>	$-2^{15}..2^{15} - 1$ ($-32768..32767$)	16 бит
<i>int</i>	$-2^{31}..2^{31} - 1$	32 бит
<i>long</i>	$-2^{63}..2^{63} - 1$	64 бит
<i>float</i>	$-3.4028235E+38..-1.4E-45$ и $1.4E-45..3.4028235E+38$	32 бит
<i>double</i>	$-1.7976931348623157E+308..-4.9E-324$ и $4.9E-324..1.7976931348623157E+308$	64 бит

Ссылочные типы и типы-значения

- ▶ Примитивные типы:
byte, short, int, long, float, double, boolean, char
- ▶ Ссылочные типы: массивы, классы (в том числе строки и типы-обёртки), интерфейсы, перечисления, аннотации
- ▶ Ссылочные типы всегда хранятся на куче и передаются по ссылке, примитивные типы всегда хранятся и передаются по значению
- ▶ У каждого типа есть значение по умолчанию — **null** для ссылочных типов (в том числе массивов и строк), нули для всех остальных
- ▶ Оператор == для ссылочных типов всегда сравнивает их место в памяти
 - ▶ Строки нельзя сравнивать ==, используйте equals

Типы-обёртки, что?

- ▶ Классы, соответствующие примитивным типам и поддерживаемые компилятором
- ▶ Boxing/Unboxing
- ▶ **byte** — Byte, **short** — Short, ну вы поняли
- ▶ Не всё так просто: **char** — Character, **int** — Integer
- ▶ Методы: **int** ololo = Integer.parseInt("239");
- ▶ Известная “особенность”:
 - ▶ Integer.valueOf(127) == Integer.valueOf(127)
 - ▶ но Integer.valueOf(128) != Integer.valueOf(128)

IntelliJ IDEA, демонстрация

Демонстрация

Что скачать и поставить

- ▶ JDK 11 (<https://www.oracle.com/technetwork/java/javase/downloads/jdk11-downloads-5066655.html>)
 - ▶ Обратите внимание, JRE — среда времени выполнения, JDK — среда времени выполнения + инструменты разработки (включая компилятор)
 - ▶ Очень желательно добавить `javac` в `PATH` и прописать переменную окружения `JAVA_HOME`
- ▶ IntelliJ IDEA (<https://www.jetbrains.com/student/>)

Что сдавать

- ▶ Файлы .java
- ▶ Папку .idea
 - ▶ Кроме workspace.xml, usage.statistics.xml, tasks.xml
- ▶ Файлы .iml (если есть)
- ▶ Ничего больше

Решения надо выкладывать на GitHub, ссылку в HwProj

Как собирать из консоли

▶ javac

- ▶ `javac MyClass.java YetAnotherClass.java`
- ▶ `javac -d classes MyClass.java`
- ▶ `javac -classpath classes;library.jar -d classes MyClass.java`

▶ CLASSPATH

- ▶ Набор путей, по которым компилятор и Java-машина ищут классы
- ▶ Всегда содержит классы из стандартной библиотеки
- ▶ По умолчанию — текущая директория (“.”)
- ▶ Задаётся как список директорий или JAR-файлов, через “;” в Windows и через “:” во всём остальном
 - ▶ JAR-файл — просто заархивированная папка с классами, по сути — библиотека

▶ Системы сборки — Gradle, Maven, ...

Как запускать из консоли

- ▶ Нет никакого .exe-шника, виртуальной машине передаётся на исполнение файл с байт-кодом класса
- ▶ Оный класс должен иметь метод
public static void main(String[] args)
- ▶ java (javaw)
 - ▶ java MyClass
 - ▶ java -classpath classes_dir;library.jar MyClass
 - ▶ java -jar library_with_main_class.jar
 - ▶ не всё так просто, jar-нику нужен манифест
 - ▶ Имя запускаемого класса должно быть **полностью квалифицированным**
 - ▶ например, *java com.example.MyClass*
 - ▶ Нелишне посмотреть документацию, есть много полезных ключей командной строки
- ▶ Системы сборки несколько облегчают эту боль

Некоторые тонкости IDEA

- ▶ Есть отдельно меню File -> Settings и отдельно File -> Project Structure
 - ▶ Settings — это в основном настройки самой среды
 - ▶ Project Structure — это настройки проекта
 - ▶ Версия языка (есть отдельно версия языка и отдельно версия SDK)
 - ▶ Модули — в какой папке код, в какой тесты, в какой ресурсы; IDEA компилирует только папки, отмеченные как Sources или Tests
- ▶ Справа сверху — конфигурации запуска. Там можно настроить, например, параметры командной строки
- ▶ Знание основных хоткеев может спасти жизнь на контрольной

Стайлгайд

- ▶ <https://google.github.io/styleguide/javaguide.html> (только для отступа используйте 4 пробела, а не 2)
- ▶ На что обратить внимание:
 - ▶ camelCase для методов и “переменных”, CamelCase для типов, КАПС_ДЛЯ_КОНСТАНТ
 - ▶ Правильные имена пакетов (DNS-имя наоборот + собственно имя пакета)
 - ▶ “Египетские” фигурные скобки (так же известны, как K & R)
 - ▶ Минимально возможная видимость полей, методов и всего-всего
 - ▶ Всегда указывайте модификатор видимости
 - ▶ Комментарии к каждому классу и каждому public-методу
 - ▶ JavaDoc

JavaDoc

- ▶ Стандартная система генерации документации
- ▶ В IDEA это Tools -> Generate JavaDoc
- ▶ `/** */` — JavaDoc-комментарий
- ▶ Сначала общее описание, затем, опционально, уточняющие тэги
- ▶ Тэги:
 - ▶ `@param` имя описание параметра
 - ▶ `@return` описание возвращаемого значения
 - ▶ `@exception` ИмяКлассаИсключения описание, когда бросается
 - ▶ `@inheritDoc`
 - ▶ `@see`
- ▶ Пустые тэги не очень полезны