ИТМО ВТ

Структура курса

- Веб-технологии
  - Технология клиент-сервер
  - Протокол HTTP
  - Клиентские технологии
    - HTML
    - CSS
    - DOM
    - JavaScript
  - Серверные технологии
    - PHP
    - **Java Enterprise Edition**
      - **Java Servlets**
      - **Java Server Pages**
      - **Java Server Faces**
- **Java Standard Edition**
- **язык программирования Java**

2 семестр

1 семестр

Программирование Интернет-приложений

- Документация — helios.cs.ifmo.ru
- Литература:
  - К. Хорстман, Г. Корнелл. Java 2. Библиотека профессионала.
  - Г. Шилдт. Java 8. Руководство для начинающих. Полное руководство
  - Д. Фланаган. Java. Справочник
  - Б. Эккель. Философия Java
  - Дж. Блох. Java. Эффективное программирование

1. Введение и синтаксис

Технология Java

- Язык программирования Java
- Платформа Java
  - JRE (Java Runtime Environment) = JVM + Java API
    - JVM — виртуальная Java-машина (HotSpot)
    - Java API — библиотека стандартных классов
- JDK (Java Development Kit) = JRE + средства разработки
- Редакции Java
  - Java SE
  - Java EE = Java SE + API для клиент-серверных бизнес-приложений
  - Java ME = подмножество Java SE для мобильных устройств

ИТМО ВТ

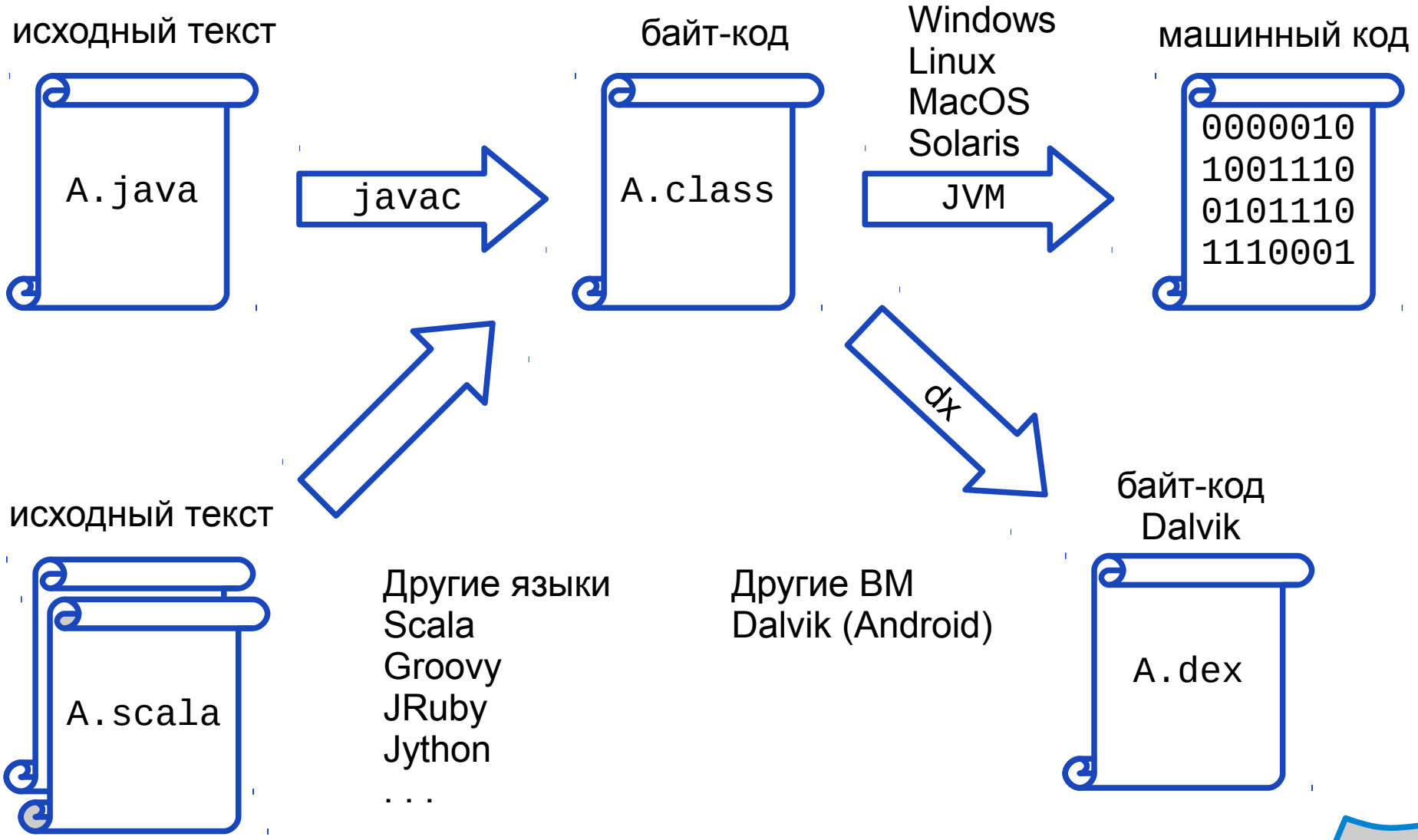
Версии Java

- Проект "Green", язык Oak
- 1996: JDK 1.0.2 — первая стабильная версия
- 1997: JDK 1.1 — внутренние классы, модель событий, рефлексия
- 1998: J2SE 1.2 (Java 2) — коллекции, Swing
- 2000: J2SE 1.3 — виртуальная машина HotSpot
- 2002: J2SE 1.4 — assert, регулярные выражения, NIO
- 2004: J2SE 5.0 — обобщенные классы, перечисления, аннотации автоупаковка, переменное число аргументов, foreach
- 2006: Java SE 6
- 2011: Java SE 7 — автозакрытие ресурсов, NIO.2, поддержка динамических языков
- 2014: Java SE 8 — лямбда-выражения, функциональные интерфейсы, методы по умолчанию, Stream API

Язык программирования Java

- Особенности:
  - Кроссплатформенность
  - Объектно-ориентированная парадигма
  - Элементы функциональной парадигмы
  - Поддержка многопоточности
  - Строгая статическая типизация
  - Автоматическое управление памятью

Язык и виртуальная машина Java



Виртуальная машина Java

- Базовый загрузчик классов (Bootstrap Class Loader)
- Системная область данных (Runtime Data Area)
  - Область методов (Method Area)
    - Пул констант (Runtime constant pool) (для каждого класса)
  - Куча (Heap) — память выделяется и освобождается динамически
  - Счетчик команд (PC register) (для каждого потока)
  - Стек (JVM Stack) (для каждого потока)
    - Фрейм (Frame) (создается при вызове метода)
      - Локальные переменные
      - Стек операндов
      - Ссылка на пул констант
- Динамический компилятор (JIT compiler)
- Сборщик мусора (Garbage collector)

ИТМО ВТ

Этапы работы виртуальной машины

- java C args
- если класс не загружен
  - загрузка класса
  - связывание класса
    - верификация (проверка байт-кода на корректность)
    - подготовка (создание статических переменных и системных таблиц)
    - разрешение (создание символических ссылок на другие классы и интерфейсы)
  - инициализация класса
    - инициализация суперклассов и интерфейсов (если они еще не загружены — перейти к их загрузке)
    - инициализация переменных и выполнение статических блоков инициализации
- запуск метода main класса C с аргументами args
- завершение работы
  - завершились все потоки не-демоны
  - был вызван метод exit() класса Runtime или System

Лексика

- Кодировка Unicode (UTF-16) — 1112064 символа
- Разделители строк — `'\n', '\r', '\r\n'`
- Разделители лексем - `' ', '\t', '\f'`
- Комментарии:

```
// однострочный комментарий
/* многострочный
   комментарий */
/** документирующий комментарий */
```

- Лексемы — идентификаторы, ключевые слова, литералы, разделители (пунктуаторы), операторы

Ключевые слова

abstract	continue	for	new	switch
assert	default	if	package	synchronized
boolean	do	goto	private	this
break	double	implements	protected	throw
byte	else	import	public	throws
case	enum	instanceof	return	transient
catch	extends	int	short	try
char	final	interface	static	void
class	finally	long	strictfp	volatile
const	float	native	super	while

Литералы

true	false	null

Идентификаторы

- не ключевое слово, не литерал
- начинается с буквы, дальше — буквы и цифры (Unicode)
- имя пакета — строчными буквами
- имя класса/интерфейса— CamelCase с заглавной буквы
- имя переменной/метода — camelCase со строчной буквы
- имя константы — SNAKE_CASE заглавными буквами
- имя переменной — существительное
- имя метода — глагол

ИТМО ВТ

Типы данных

- Примитивные
- хранится значение
  - byte
  - short
  - int
  - long
  - float
  - double
  - boolean
  - char
- Ссылочные
- хранится ссылка на значение
  - Массивы
  - Классы
    - Перечисления
  - Интерфейсы
    - Аннотации

ИТМО ВТ

Целые типы и литералы

- Типы со знаком:
  - byte (1 байт)
  - short (2 байта)
  - int (4 байта)
  - long (8 байтов)
- Литералы
  - десятичный
    - 123456789
  - восьмеричный
    - 01234567
  - шестнадцатеричный
    - 0xabcdef
  - двоичный
    - 0b1111_0000_1010_0101
  - по умолчанию — int
  - суффикс L — long
    - 9_876_543_210L

ИТМО ВТ

Типы и литералы с плавающей запятой (точкой)

- Типы стандарта IEEE 754:
  - float (32 бита)
  - double (64 бита)
  - ошибки округления!
  - ± 0
  - ± бесконечность (1.0/0.0)
  - не число (NaN) (0.0/0.0)
- Литералы
  - десятичный
    - 3.1415926
    - 0.0
    - 6.626e-34 = 6.626×10^{-34}
  - шестнадцатеричный
    - 0xabc.def
    - 0xaP4 = $a_{16} \times 2^4$
  - по умолчанию — double
  - суффикс F — float
    - 9.7531F

ИТМО ВТ

Логический тип и литералы

- Тип boolean
  - на уровне языка не int
  - в виртуальной машине - int
- Литералы
  - true
  - false

ИТМО ВТ

Символьный тип и литералы

- Тип char (без знака)
  - 2 байта
  - кодировка Unicode
- Тип String не примитивный
- Символьные литералы
  - `'a'` (латинские символы)
  - `'ы'` (символы национальных алфавитов)
  - `'\u0234'` (символы Unicode)
  - `'\024'` (восьмеричный код)
  - `'\n'` (перевод строки)
  - `'\t'` (табуляция)
  - `'\\'` (обратный слэш)
  - `'\''` (одинарная кавычка)
  - `'\"'` (двойная кавычка)
- Строковые литералы
  - `"Hello, world!\n"`

Переменные

- объявление (declaration) — выделение памяти
  - *[модификаторы] тип идентификатор*;
  - int a; // (4 байта)
- инициализация (initialization) — присваивание значения
  - a = 25;
- объявление с инициализацией
  - int a = 25;
- константы
  - final int b = 50;

Операторы

- Приоритет
  - 2 + 2 * 2
- Ассоциативность
  - L: a + b — c → (a + b) — c
  - R: a = b = 2 + 3 → a = (b = 2 + 3)
- Количество и тип операндов
- Тип результата
- Побочные эффекты

Приоритет и ассоциативность операторов

- 16L: `o.x  o::x  x[]  m()  i++  i--`
- 15R: `++i  --i  +x  -x  ~  !`
- 14R: `new  (t)x`
- 13L: `*  /  %`
- 12L: `+  -`
- 11L: `<<  >>  >>>`
- 10L: `<  <=  >  >=  instanceof`
- 9L: `==  !=`
- 8L: `&` 7L: `^` 6L: `|`
- 5L: `&&` 4L: `||`
- 3R: `? :`
- 2R: `=  *=  /=  %=  +=  -=  <<=  >>=  >>>=  &=  ^=  |=`
- 1R: `->`

ИТМО ВТ

Операторы инкремента

- Имеют побочный эффект — изменение операнда
- Постфиксные (изменение после получения значения)

```
int a;
int i = 1;
a = i++;
```

  - a = 1, i = 2
- Префиксные (изменение до получения значения)

```
int a;
int i = 1;
a = ++i;
```

  - a = 2, i = 2

ИТМО ВТ

Преобразование примитивных типов

- byte < short / char < int < long < float < double
- расширяющая конверсия — разрешена

  int a = 2;

  long b = a;
- сужающая конверсия — только явно

  int a = 2;

  byte b = a; // ошибка компиляции

  byte b = (byte) a;
- возможна потеря точности
  - (float) int
  - (float) long
  - (double) long
- двунаправленная конверсия
- (char) byte = (int) byte; (char) int

ИТМО ВТ

Арифметические операторы

- + - * / %
- % - остаток при делении
- операнды приводятся к int, long, float, double
  - 5 / 2 = 2
  - 5 % 2 = 1
  - 5 / 2.0 = 2.5
  - 5 / 0 = исключение
  - 5 / 0.0 = ∞

```
byte a = 1;
byte b = 1;
byte c = a + b; // ошибка
byte c = (byte) a + b;
```

- + для строк — конкатенация
  - "hello" + "hello" = "hellohello"

Операторы сдвига

- Знаковый сдвиг
  - `<< >>`
- Беззнаковый сдвиг
  - `>>>`

```
0b1111...1111 >> 1   = 0b1111...1111
0b1111...1111 >>> 1  = 0b0111...1111
```

Логические и побитовые операторы

- `~ & ^ |`
  - для типа boolean — логические NOT, AND, XOR, OR
    - true & false = false
  - для целых типов — побитовые
    - `0b1100 & 0b1010  = 0b1000`
- `&& ||`
  - только для типа boolean
  - второй операнд не вычисляется, если это не надо
  - `false &&   - false`
  - `true  ||   - true`
  - `if (a != 0 && x / a > 5)`

Условный оператор

- правая ассоциативность
- a = x < 0 ? -x : x; // a – модуль x
- x (0 - рубль, 1 - рубля, 2 - рублей)

```
int plural =
  x % 10 == 1 && x % 100 != 11
    ? 0
    : x % 10 >= 2 && x % 10 <= 4 &&
      (n % 100 < 10 || n % 100 >= 20)
        ? 1
        : 2;
```

Операторы присваивания

- Правая ассоциативность

- a += b
- a = a + b

- a >>= 4
- a = a >> 4

Выражения, инструкции, блоки

- Выражения

  Состоят из переменных, литералов и операторов

  - a + 2 * 4 << 2

  - Имеют значение

- Инструкции — базовые элементы программы
  - Завершаются символом ;
  - Не имеют значения
- Блок — объединение инструкций фигурными скобками { }
- Инструкция присваивания:
  - *переменная* = *выражение*;
- Управляющие инструкции

if-then-else

```
if (условие) {
    действия при выполнении условия
} else {
    действия при невыполнении условия
}
```

switch-case-default

```
switch (выражение) {
    case значение1:
        действия, если выражение имеет значение1
        break;
    case значение2:
        действия, если выражение имеет значение2
        break;
    default:
        если выражение имеет другое значение
}
```

- тип выражения – byte, short, char, int, enum, String

while и do

```
while (условие) {
    повторяющееся действие
}

do {
    повторяющееся действие
} while (условие)
```

for

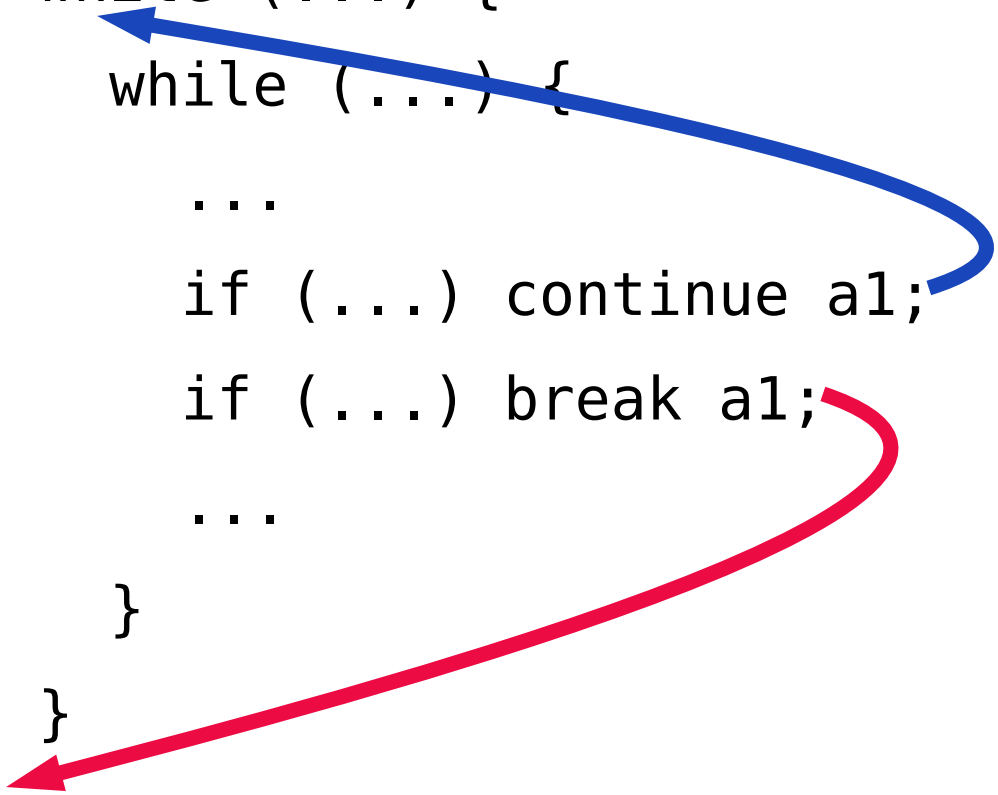
```
for (инициализатор; условие; итератор) {
    повторяющееся действие
}

инициализатор;
while (условие) {
    повторяющееся действие
    итератор;
}
```

break-continue

- break — выйти из цикла
- continue — продолжить цикл, не завершая итерацию
- могут использоваться с метками для выхода из вложенных циклов

```
a1: while (...) {
      while (...) {
        ...
        if (...) continue a1;
        if (...) break a1;
        ...
      }
    }
```



Ссылочные типы

- Классы
- Интерфейсы
- Массивы

Массивы

- объявление (declaration) — память под ссылку
  - int[] b; // (4 байта) — ссылка на массив
- создание (creation) — память под элементы массива
  - b = new int[5]; // 5 * 4 байта = 20 байтов
- инициализация (initialization) — присвоение значений
  - b[0] = 30; b[1] = 44;
- объявление с инициализацией
  - int[] b = { 1, 2 ,4, 8, 16 };
- количество элементов массива
  - b.length

Многомерные массивы

- Многомерный массив — массив массивов

```
int[][] a;
a = new int[2];
a[0] = new int[2];
a[1] = new int[3];
a[0][1] = 10; a[0][2] = 20;
a[1][1] = 15; a[1][1] = 25; a[1][2] = 35;
float[][] b = {
   { 0.1, 0.2, 0.3 },
   { 1.1, 2.2, 3.3, 4.4 }
}
```

Инструкция foreach

```
int[] a;
a = new int[10];
for(int i = 0; i < 10; i++) {
    a[i] = i;
}
for (int x : a) {
    System.out.println(x);
}
```