



Лекция 2. Синтаксис Java. Простые типы данных.

NetCracker®

```

class ПриветМир {
    public static void main(String[] args) {
        ПриветМир мир = new ПриветМир();
        мир.привет();
    }
    public void привет() {
        System.out.println("Привет!");
        int a = 0;
        \u0061 = 10; // 0x0061 - Unicode код символа 'a'
        System.out.println("\u0061 = " + a);
        System.out.println("\uu0061 \uuu0061 \u000D \n");
    }
}

```

- Java ориентирован на Unicode
- Первые 128 символов почти идентичны набору ASCII
- Символы Unicode задаются с помощью escape-последовательностей
 \u262f, \uu2042, \uuu203d

Таблицы кодов ASCII

Таблица 1

Коды управляющих символов (0–31)				
Код	Обозначение	Клавиша	Значение	Отображаемый символ
1	2	3	4	5
0	nul	^@	Нуль	
1	soh	^A	Начало заголовка	☺
2	stx	^B	Начало текста	☼
3	etx	^C	Конец текста	♥
4	eot	^D	Конец передачи	♦
5	enq	^E	Запрос	♣
6	ack	^F	Подтверждение	♠
7	bel	^G	Сигнал (звонок)	•
8	bs	^H	Забой (шаг назад)	▣
9	ht	^I	Горизонтальная табуляция	○
10	lf	^J	Перевод строки	▣
11	vt	^K	Вертикальная табуляция	♂
12	ff	^L	Новая страница	♀
13	cr	^M	Возврат каретки	♪
14	so	^N	Выключить сдвиг	🎵
15	si	^O	Включить сдвиг	☀
16	dle	^P	Ключ связи данных	▶
17	dc1	^Q	Управление устройством 1	◀
18	dc2	^R	Управление устройством 2	↕
19	dc3	^S	Управление устройством 3	!!
20	dc4	^T	Управление устройством 4	¶
21	nak	^U	Отрицательное подтверждение	§
22	syn	^V	Синхронизация	—
23	etb	^W	Конец передаваемого блока	↕
24	can	^X	Отказ	↑
25	em	^Y	Конец среды	↓
26	sub	^Z	Замена	→
27	esc	^[Ключ	←
28	fs	^\	Разделитель файлов	L
29	gs	^]	Разделитель группы	↔
30	rs	^^	Разделитель записей	▲
31	us	^_	Разделитель модулей	▼

Примечание. В графе «Клавиши» обозначение ^ соответствует нажатию клавиши <Ctrl>, вместе с которой нажимается соответствующая «буквенная» клавиша, формируя код символа.

Таблица 2

Символы с кодами 32–127							
Код	Символ	Код	Символ	Код	Символ	Код	Символ
32	пробел	56	8	80	P	104	H
33	!	57	9	81	Q	105	I
34	"	58	:	82	R	106	J
35	#	59	;	83	S	107	K
36	\$	60	<	84	T	108	L
37	%	61	=	85	U	109	M
38	&	62	>	86	V	110	N
39	'	63	?	87	W	111	O
40	(64	@	88	X	112	P
41)	65	A	89	Y	113	Q
42	*	66	B	90	Z	114	R
43	+	67	C	91	[115	S
44	,	68	D	92	\	116	T
45	-	69	E	93]	117	U
46	.	70	F	94	^	118	V
47	/	71	G	95	_	119	W
48	0	72	H	96	`	120	X
49	1	73	I	97	A	121	Y
50	2	74	J	98	b	122	Z
51	3	75	K	99	c	123	{
52	4	76	L	100	d	124	
53	5	77	M	101	e	125	}
54	6	78	N	102	f	126	~
55	7	79	O	103	g	127	del

Таблица символов Юникода

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0000																
0010																
0020		!	"	#	\$	%	&	'	()	*	+	,	-	.	/
0030	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
0040	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
0050	P	Q	R	S	T	U	V	W	X	Y	Z	[\]	^	_
0060	`	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
0070	p	q	r	s	t	u	v	w	x	y	z	{		}	~	

Компилятор, анализируя программу, сразу разделяет ее на

- пробелы (white spaces);
- комментарии (comments);
- основные лексемы (tokens).

Виды лексем

- идентификаторы (identifiers);
- ключевые слова (key words);
- литералы (literals);
- разделители (separators);
- операторы (operators).

Литералы

- целочисленный (integer);
- дробный (floating-point);
- булевский (boolean);
- символный (character);
- строковый (string);
- null-литерал (null-literal).

```

class CommentExample {
    int number = 42; // Комментарии
    // могут быть однострочными

    /* А могут быть
       и многострочными.
       int oldWrongCode = 0;
    */

    /**
     * А это javadoc к методу, который
     * возвращает 1 деленную на число.
     * Комментарий, включающий информацию о:
     * <ol>
     * <li>Классах и интерфейсах</li>
     * <li>Методах</li>
     * <li>Полях</li>
     * </ol>
     * Может содержать ссылки:
     * {@link CommentExample#number},
     * а так же специальные теги:
     * @author Dsiuba Aleksey <Dsiuba@NetCracker.com>
     * @param x число, которое нужно обратить
     * @return число, обратное к x
     * @throws ArithmeticException если x равен нулю
     * @see java.lang.Double
     * @deprecated
     */
    public double reverse(double x) {
        return 1 / x;
    }
}

```

The screenshot shows the Javadoc for the `reverse` method in the `CommentExample` class. The Javadoc content is as follows:

```

/**
 * А это javadoc к методу, который
 * возвращает 1 деленную на число.
 * Комментарий, включающий информацию о:
 * <ol>
 * <li>Классах и интерфейсах</li>
 * <li>Методах</li>
 * <li>Полях</li>
 * </ol>
 * Может содержать ссылки:
 * {@link CommentExample#number},
 * а так же специальные теги:
 * @author Dsiuba Aleksey <Dsiuba@NetCracker.com>
 * @param x число, которое нужно обратить
 * @return число, обратное к x
 * @throws ArithmeticException если x равен нулю
 * @see java.lang.Double
 * @deprecated
 */

```

The IDE interface shows the following sections:

- Field Summary:** Shows the `number` field with type `int`.
- Constructor Summary:** Shows the `CommentExample()` constructor.
- Method Summary:** Shows the `reverse(double x)` method with the `Deprecated` tag.
- Methods inherited from class java.lang.Object:** Lists `clone`, `equals`, `finalize`, `hashCode`, `notify`, `notifyAll`, `wait`, `wait(long)`, `wait(Timeout)`.
- Field Detail:** Shows the `number` field with type `int`.
- Constructor Detail:** Shows the `CommentExample()` constructor.
- Method Detail:** Shows the `reverse` method with the `Deprecated` tag.
- Comments:** Shows the full Javadoc content.

Операторы

= > < ! ? : == && || .

Литералы

Значения, заданные в коде программы

Разделители

{ } [] () ; ,

final double number = sqrt ("Word");

Служебные слова

abstract, continue, for, new,
switch, assert, default, goto, package,
synchronized, boolean, do, if, 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

Идентификаторы

Имена программных сущностей:
классов, полей, методов,
переменных

- Первый символ должен быть буквой, подчеркиванием (_) или знаком доллара(\$).
- Среди остальных символов могут быть также и цифры.
- Идентификаторы чувствительны к регистру букв.
- Нельзя использовать зарезервированные слова.

Верные идентификаторы:

```
MyClass1
$amount
totalGrades
TotalGrades
TAX_RATE
one
```

Неверные идентификаторы:

```
My Class    пробел в имени
numberOf*s  звездочка
final       ключевое слово
1Way        недопустимое начало
```

```

boolean истина = true;
boolean ложь = false;

/** 16-битный символ Unicode */
char символ = 'a';
char спецсимвол = '\n';
char слэш = '\\';
char восьмеричныйКод = '\377';
char шестнадцатеричныйКод = '\uFFFF';
// \t, \b, \n, \r, \f, \', \", \\

/** 8-битное целое со знаком */
byte минимальный = -128;
byte максимальный = 0xFF; // +127

/** 16-битное целое со знаком */
short число = 300;
short шестнадцатеричное = 0xB0bE;

/** 32-битное целое со знаком */
int восьмеричное = 013; // = 12;

/** 64-битное целое со знаком */
long большое = 9223372036854775807L;
long сУказаниемТипа = 8L;

/** 32-битное с плавающей точкой */
float целое = 12;
float дробное = 3.14;
float однадесятая = .1;
float один = 1.;
float сТипом = 0f;
float экспоненциальное = 1e3;
// 1*10^3 = 1000
float ещеОдно = 2E-3;
// 0.002

/** 64-битное с плавающей точкой */
double простое = 2.1;
double сУказаниемТипа = 0.00d;

/** Строка - набор символов */
String строка =
    "То же, что и в \"char\"\n";

```


Любая переменная имеет три характеристики:

- ИМЯ
- ТИП
- значение

Переменная может иметь инициализаторы:

```
int a;  
int b = 3 + 2;  
int c = b + 2;  
int d = a = c;
```

Ключевое слово *final*

```
final double g = 9.81;
```

Тип	Длина (байты)	Диапазон значений
byte	1	-128 ... 127
short	2	-32,768 ... 32,767
int	4	-2,147,483,648 ... 2,147,483,647
long	8	-9,223,372,036,854,775,808 ... 9,223,372,036,854,775,807
char	2	'\u0000' ... '\uffff', или 0 ... 65,535

- **операции сравнения** (возвращают булево значение)
 - <, <=, >, >=
 - ==, !=
- **числовые операции** (возвращают числовое значение)
 - унарные операции + и -
 - арифметические операции +, -, *, %
 - **деление /**
 - операции инкремента и декремента (в префиксной и постфиксной форме): ++ и --
 - **операции битового сдвига** <<, >>, >>>
 - **битовые операции** ~, &, |, ^
- **оператор с условием** condition ? true_value : false_value
- **оператор приведения типов** (type) value
- **оператор конкатенации со строкой** +

```
int x = -2147483648; // Наименьшее возможное типа int
int y = -x;
System.out.println(y); // Результат: -2147483648
```

```
x = 300000;
System.out.println(x * x); // -194.313.216
```

```
      300.000
*      300.000
= 90.000.000.000
```

```
b 1.0100.1111.0100.0110.1011.0000.0100.0000.0000
i      0000.1011.1001.0100.1111.1011.1111.1111
+      0000.0000.0000.0000.0000.0000.0000.0001
=      0000.1011.1001.0100.1111.1100.0000.0000
```

```
= -194.313.216
```

Название	Длина (байты)	Диапазон значений
float	4	$3.402,823,47 \times 10^{38}$ $1.402,398,46 \times 10^{-45}$
double	8	$1.797,693,134,862,315,70 \times 10^{308}$ $4.940,656,458,412,465,44 \times 10^{-324}$

Операция	Результат
1.0/0.0	Double.POSITIVE_INFINITY $+\infty$
-1.0/0.0	Double.NEGATIVE_INFINITY $-\infty$
0.0/0.0	Double.NaN Не-число
(1.0/0.0)*0.0	Double.NaN Не-число
-0.0	Отрицательный ноль

Пример overflow:

```
1e20f * 1e20f = Infinity  
-1e200 * 1e200 = -Infinity
```

Пример underflow:

```
System.out.println( 1e-40f / 1e10f);  
System.out.println(-1e-300 / 1e100);  
float f = 1e-6f;  
System.out.println(f);  
f += 0.002f;  
System.out.println(f);  
f += 3;  
System.out.println(f);  
f += 4000;  
System.out.println(f);
```

```
0.0  
-0.0  
  
0000.000001  
  
0000.002001  
  
0003.002001  
  
4003.002000
```

Файловая система: \src\ua\edu\sumdu\j2se\pr1\MainClass.java

<code>package ua.edu.sumdu.j2se.pr1;</code>	Пакет текущего файла
<code>import ua.edu.sumdu.j2se.pr1.operations.Operation;</code>	Импорт внешнего класса
<code>public class MainClass {</code>	Объявление класса
<code> public static void main(String[] args) {</code>	Объявление метода
<code> for (int x = 1; x < 10; x++) {</code>	Тело метода
<code> for (int y = 1; y < 10; y++) {</code>	
<code> <u>Operation</u> operation = new <u>Operation</u>(x,y);</code>	
<code> System.out.printf("%3d", operation.getResult());</code>	
<code> }</code>	
<code> System.out.println();</code>	
<code> }</code>	
<code> }</code>	
<code> public static final String TITLE = "Hello, World!";</code>	Объявление поля
<code>}</code>	

```
// предшествующая часть блока
{ // начало блока
    ...
    int x = 1; // тут существует x
    { // еще один блок
        ...
        int y; // тут существует y
        ...
    } // переменная y уничтожается
    ...
    x += y; // ошибка!
}
```



```
class Echo {  
    public static void main(String[] args) {  
        for (int i = 0; i < args.length; i++)  
            System.out.print(args[i] + " ");  
            System.out.println();  
        }  
    }
```

- предоставляют необходимую информацию для компилятора;
- предоставляют метаданные различным инструментам для генерации кода, конфигураций и т.д.;
- использоваться в коде во время выполнения программного кода (reflection).

```
class Parent {  
    public void doSomething() { }  
}  
  
public class Main extends Parent {  
    @Override  
    public void doSomething() { }  
}
```

•Синтаксис Java. Основы языка. Основные конструкции:

<http://www.intuit.ru/department/pl/javapl/>

<http://www.javapassion.com/javaintro/>

•Список зарезервированных слов Java

http://java.sun.com/docs/books/tutorial/java/nutsandbolts/_keywords.html

Список литературы



Q&A



A background image showing a complex network of white lines and nodes on a blue gradient. The nodes are represented by small white circles, and the lines connect them in a web-like pattern, suggesting a network or data flow.

Thank you!

