Pemrograman Berbasis Obyek

Dasar Pemrograman Java

Oleh Politeknik Elektronika Negeri Surabaya
2017

Politeknik Elektronika Negeri Surabaya
Departemen Teknik Informatika dan Komputer

Konten

- Identifier
- Java Keywords
- Primitive Types
- Literals
- Conversion of Primitive

Identifier

- Nama yang digunakan oleh programer untuk memberi nama pada variable, class, atau method.
- Tidak boleh mengandung spasi
- Can start with a Unicode letter, underscore (_), or dollar sign ($)
- Are case-sensitive and have no maximum length
- Examples:

```
1. foobar                  // legal
2. BIGinterface            // legal: embedded keywords
3.                         // are OK.
4. $incomeAfterExpenses      // legal
5. 3_node5                 // illegal: starts with a digit
6. !theCase                // illegal: must start with
7.                         // letter, $, or _
```

Java KeywordS

- are considered as reserved keywords
- may not be used as identifiers.
- None of the reserved words have a capital letters
- 2 keywords that are reserved in Java but which are not used : const dan goto

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

- ** added in 1.2
- *** added in 1.4
- **** added in 5.0

Primitive Types

- The Java programming language defines eight primitive types:
  - Logical - `boolean`
  - Textual - `char`
  - Integral - `byte`, `short`, `int`, and `long`
  - Floating - `float` and `double`

Type	Bits size	Value limit	Type	Bits size	Value Limit
boolean	1	true, false	char	16	
byte	8	$-2^7 \rightarrow 2^7 - 1$	short	16	$-2^{15} \rightarrow 2^{15} - 1$
int	32	$-2^{31} \rightarrow 2^{31} - 1$	long	64	$-2^{63} \rightarrow 2^{63} - 1$
float	32		double	64	

Literals

- Literals adalah value
- Tidak bias dituliskan di sisi kiri assignments “ = ”
- Contoh
  - boolean result = true;
  - char capitalC = 'C';
  - byte b = 100;
  - short s = 10000;
  - int i = 100000;

Literals

Name	Type	Value
Logical	Boolean	True, false
Textual / Char	char	Unicode character (16-bit encoding → 0 s/d 2^{16} - 1)
String	String	"sequence of character"
Integral	byte, short, int, long	Decimal, Octal, Hexadecimal **2** The decimal value is 2 **077** The leading 0 indicates an octal value **0xBAAC** The leading 0x indicates a hexadecimal **100L** The "L" or "l" suffix indicate a Long integer
Floating point	Float, double	3.14 → a simple floating point value (a double) 6.02E23 → a large floating point value ("E") 2.718F → a simple float size value ("F" float) 123.4E306D → a large double value ("D" double)

Underscore in Numeric Literals

- In Java SE 7 and later, any number of underscore characters (_) can appear anywhere between digits in a numerical literal. This feature enables you, for example. to separate groups of digits in numeric literals, which can improve the readability of your code.
- For instance, if your code contains numbers with many digits, you can use an underscore character to separate digits in groups of three, similar to how you would use a punctuation mark like a comma, or a space, as a separator.
  - long creditCardNumber = 1234_5678_9012_3456L;
  - float pi = 3.14_15F;
  - long hexBytes = 0xFF_EC_DE_5E;
  - long maxLong = 0x7fff_ffff_ffff_ffffL;
  - byte nybbles = 0b0010_0101;

Underscore in Numeric Literals

- You can place underscores only between digits
- you cannot place underscores in the following places:
  - At the beginning or end of a number
  - Adjacent to a decimal point in a floating point literal
  - Prior to an F or L suffix
  - In positions where a string of digits is expected

Contoh

- **// Invalid: cannot put underscores**
- **// adjacent to a decimal point**
- float pi1 = 3_.1415F;

- **// Invalid: cannot put underscores**
- **// adjacent to a decimal point**
- float pi2 = 3._1415F;

- **// Invalid: cannot put underscores**
- **// prior to an L suffix**
- long socialSecurityNumber1 = 999_99_9999_L;

- // This is an identifier, not a numeric literal
- int x1 = _52;

- // OK (decimal literal)
- int x2 = 5_2;

- **// Invalid: cannot put underscores**
- **// At the end of a literal**
- int x3 = 52_;

- // OK (decimal literal)
- int x4 = 5_____2;

- **// Invalid: cannot put underscores**
- **// in the 0x radix prefix**
- int x5 = 0_x52;

- **// Invalid: cannot put underscores**
- **// at the beginning of a number**
- int x6 = 0x_52;

- // OK (hexadecimal literal)
- int x7 = 0x5_2;

- **// Invalid: cannot put underscores**
- **// at the end of a number**
- int x8 = 0x52_;

Conversion of primitives

- Terjadi pada saat kompile.
- Conversion of a primitives bisa terjadi pada:
  - Assignment (widening conversion)
  - Casting (narrowing conversion)
  - Method call
  - Arithmetic promotion

Conversion of primitives (Assignment)

- Terjadi ketika suatu nilai kita berikan pada suatu variabel yang tipe datanya berbeda dari data aslinya.
- Tipe data yang baru harus mempunyai ukuran lebih besar dari tipe data yang lama.

```
int i;
double d;
i = 10;
d = i; // Assign an int value to a double variable
```

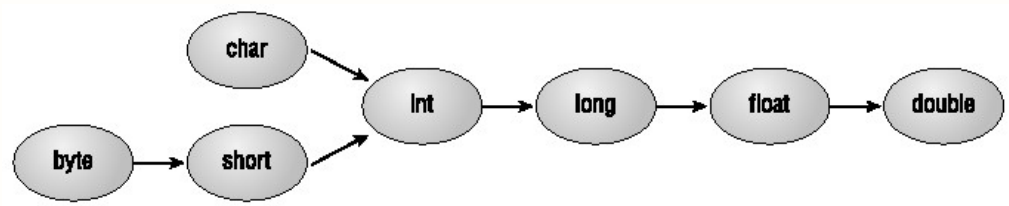
Conversion of primitives (Assignment)

- Boolean tidak bisa di konversi ke tipe data lain
- Non-boolean dapat di konversi ke tipe data lain selain Boolean
- konversi yang dilakukan adalah ***widening conversion (promotion)***

Note: *widening conversion* adalah merubah tipe data suatu variabel ke tipe data yang ukuran bit nya lebih besar dari aslinya.

Conversion of primitives (Assignment)

- ***widening conversion (promotion)***
  - From a byte to a short, an int, a long, a float, or a double
  - From a short to an int, a long, a float, or a double
  - From a char to an int, a long, a float, or a double
  - From an int to a long, a float, or a double
  - From a long to a float or a double
  - From a float to a double



- **Note:** Konversi antar primitive types yang tidak mengikuti arah panah disebut dengan *narrowing conversion*.

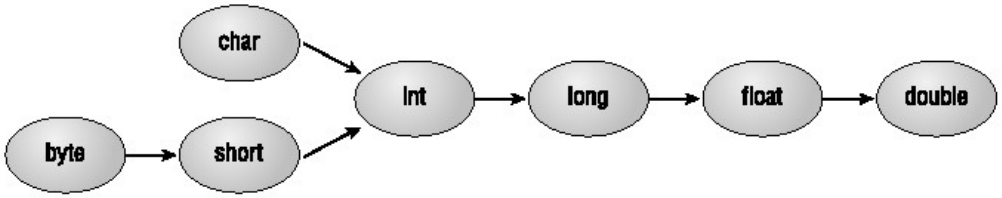
Conversion of primitives (Casting)

- *Casting* means explicitly telling Java to make a conversion.
- Dilakukan dengan cara menambahkan tipe data yang diinginkan dalam tanda kurung sebelum nilai.
- Dilakukan ketika suatu nilai kita berikan pada suatu variabel yang tipe datanya berbeda dari data aslinya. Dimana tipe data yang baru mempunyai ukuran lebih kecil dari tipe data yang lama.

```
int i = 5;
double d = (double) i;
float f = 9;
double db = (double) f;
```

Conversion of primitives (Casting)

- ***narrowing conversion (casting)***
  - From a byte to a char
  - From a short to a byte or a char
  - From a char to a byte or a short
  - From an int to a byte, a short, or a char
  - From a long to a byte, a short, a char, or an int
  - From a float to a byte, a short, a char, an int, or a long
  - From a double to a byte, a short, a char an int, a long, or a float



- **Note:** Ubah /balik arah panah.

Conversion of primitives (Method call)

- Terjadi ketika kita berusaha melewatkan suatu nilai variabel sebagai argumen suatu method, dimana tipe data variabel method tersebut berbeda dengan yang diterima.

```
1. float frads;
2. double d;
3. frads = 2.34567f;
4. d = Math.cos(frads); // Pass float to method
                        // that expects double
```

- Hint: `Math.cos(double d);`
- Pada contoh diatas frands yang bertipe float akan secara otomatis di konversi menjadi double.
- Pada contoh diatas terjadi widening conversions.

Conversion of primitives (Arithmetic conversion)

- Terjadi pada operasi matematika.
- Menggunakan operators unary : +, -, ++, --, ~
- Atau operators binary : +, -, *, /, %, >>, >>>, <<, &, ^, |
- Kompiler berusaha mencari tipe data yang sesuai dengan tipe data operan yang berbeda-beda.
  - Jika salah satu operan adalah double, operan lain dikonversikan ke double.
  - Jika salah satu operan adalah float, operan lain dikonversikan ke float.
  - Jika salah satu operan adalah long, operan lain dikonversikan ke long.
  - Selain tipe data diatas maka dikonversikan ke int.

Tugas

1. Buatlah uraian yang berisi tentang spesifikasi 8 tipe data dasar !
2. Apakah yang dimaksud dengan casting (narrowing conversion) ?
3. Apakah yang dimaksud dengan konversi (widening conversion) ?

1. Oracle Java Documentation, The Java™ Tutorials, https://docs.oracle.com/javase/tutorial/, Copyright © 1995, Oracle 2015.
2. Tita Karlita, Yuliana Setrowati, Rizky Yuniar Hakkun, Pemrograman Berorientasi Obyek, PENS-2012
3. Sun Java Programming, Sun Educational Services, Student Guide, Sun Microsystems, 2001.
4. John R. Hubbard, Programming With Java, McGraw-Hill, ISBN: 0-07-142040-1, 2004.
5. Patrick Niemeyer, Jonathan Knudsen, Learning Java, O'reilly, CA, ISBN: 1565927184, 2000.
6. Philip Heller, Simon Roberts, Complete Java 2 Certification Study Guide, Third Edition, Sybex, San Francisco, London, ISBN: 0-7821-4419-5, 2002.
7. Herbert Schildt, The Complete Reference, Java™ Seventh Edition, Mc Graw Hill, Osborne, ISBN: 978-0-07-163177-8, 2007