Language Specification – Java

Table of Contents

Language Specification – Java

Data types

Data Type	Bytes	Range
Integer types		
byte	1	-128 to 127
short	2	-32,768 to 32,767
int	4	-2,147,483,648 to 2,147,483,647
long	8	-9,223,372,036,854,775,808 to 9,223,372,036,854,775,807
Real-number types		
float	4	-3.4^{38} to -1.4^{-45}, 0, 1.4^{-45} to 3.4^{38}
double	8	-1.8^{308} to -4.9^{-324}, 0, 4.9^{-324} to 1.8^{308}
Other types		
boolean	~1 bit	true or false
char	2	A Unicode character, from 0 to 65,535 (or hex FFFF)
String	variable	A sequence of 0 to 2,147,483,647 characters

String and character literals

String literal

● A string literal is a sequence of zero or more characters enclosed in double quotes.

Character literal

● A character literal is one character enclosed in single quotes.

Identifier rules

● An identifier:
- ✓ Begins with a letter, dollar sign, or underscore.
- ✓ Is followed by zero or more letters, digits, dollar signs, and underscores.

Variable declarations

```
<data-type> <variable-name>;
OR
<data-type> <variable-name> = <initial-value>;
```

Constant declarations

```
final <data type> <constant-name> = <value>;
```

Language Specification – Java

Array declarations

```
<data-type>[] <array-name>;
OR
<data-type>[] <array-name> =
     new <data-type>[<integer-expression>];
OR
<data-type>[] <array-name> = {<initializer-list};
```

Other declarations

Array list

```
ArrayList<data-type> <arraylist-name> =
     new ArrayList<data-type>();
OR
ArrayList<data-type> <arraylist-name> =
     new ArrayList<data-type>(<integer-expression>);
```

The angle brackets around data-type are required.

Keyboard input

```
Scanner <keyboard-name> = new Scanner(System.in);
<string-variable> = <keyboard-name>.next();
OR
<integer-variable> = keyboard.nextBoolean();
OR
<integer-variable> = keyboard.nextByte();
OR
<double-variable> = <keyboard-name>.nextDouble();
OR
<integer-variable> = keyboard.nextFloat();
OR
<integer-variable> = keyboard.nextInt();
OR
<string-variable> = <keyboard-name>.nextLine();
OR
<integer-variable> = keyboard.nextLong();
OR
<integer-variable> = keyboard.nextShort();
(and others)
```

Language Specification – Java

Screen output

```
System.out.print(<expression>);
System.out.println(<expression>);
System.out.printf(<string-expression-with-format-specifiers>,
<expression-list>);
```

Escape sequences

Sequence	Purpose
\\	Output a backslash.
\’	Output a single quote.
\”	Output a double quote.
\b	Backspace cursor.
\f	Move cursor to start of new page.
\n	Start on a new line.
\r	Move cursor to start of current line.
\t	Tab on the current line.

Assignment statements

```
<variable> = <expression>;
```

Arithmetic operators

Operator	Purpose
+	Addition
–	Subtraction
*	Multiplication
/	Division
%	Modulo (integer remainder)

Language Specification – Java

Relational operators

Operator	Purpose
==	Equal to
!=	Not equal to
<	Less than
<=	Less than or equal to
>	Greater than
>=	Greater than or equal to

Logical operators

Operator	Purpose
!	Not
&&	And
||	Or

Conditionals

if statement

```
if (<condition>)
{
     <block>
}
else if (<condition>)
{
     <block>
}
else
{
     <block>
}
```

<block> is one or more statements. If there is only one statement, then the curly braces may be omitted.

Language Specification – Java

switch statement

```
switch (<expression>)
{
      case <literal-1>:
            <block>;
            break;
      case <literal-2>:
            <block>;
            break;
      …
      case <literal-n>:
            <block>;
            break;
      default:
            <block>;
}
```

<block> is one or more statements.

Iteration

while statement

```
while (<condition>)
{
      <block>
}
```

do-while statement

```
do
{
      <block>
}
while (<condition>);
```

for statement

```
for (<initialization>; <condition>; <update>)
{
      <block>
}
```

Language Specification – Java

for-each statement

```
for (<data-type> <variable-name> : <collection>)
{
     <block>
}
```

<block> is one or more statements. If there is only one statement, then the curly braces may be omitted.

Reserved words

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