Java: Primitive data types

Where are eight primitive data types in Java:

size	type	type
8	byte	boolean
16	short	char
32	int	float
64	long	double

Values of class type are references.

Type	Description	Default	Size	Example Literals
boolean	true or false	false	1* bit	true, false
byte	integer	0	8 bits	(none)
char	Unicode character	\u0000	16 bits	'a', '\u0041', '\101', '\\', 1, '\n', 'ß'
short	integer	0	16 bits	(none)
int	integer	0	32 bits	-2, -1, 0, 1, 2, 0x1A, 0b1, 017, 1_000
long	integer	0	64 bits	-2L, -1L, 0L, 1L, 2L
float	IEEE 754 floating point	0.0	32 bits	1.23e100f, -1.23e-100f, .3f, 3.14F
double	IEEE 754 floating point	0.0	64 bits	1.2345, -1.23456e-300, 1e1

int a; // creation (allocation of 4 bits in the memory) for a new integer with value 0 (default value).

byte b = 1; // 1 is a literal for an integer

Boolean

true / false

Character

The 16-bit Unicode character (UTF-16)

Numeric

Range of numeric data types in Java

Type	Size	Range
byte	8 bits	-128 .. 127
short	16 bits	-32,768 .. 32,767
int	32 bits	-2,147,483,648 .. 2,147,483,647
long	64 bits	-9,223,372,036,854,775,808 .. 9,223,372,036,854,775,807
float	32 bits	$3.40282347 \times 10^{38}$, $1.40239846 \times 10^{-45}$
double	64 bits	$1.7976931348623157 \times 10^{308}$, $4.9406564584124654 \times 10^{-324}$

Conversions: Widenings and Narrowings

		from						
to		byte	short	char	int	long	float	double
byte		[id]	🟥	🟥	🟥	🟥	🟥	🟥
short		🟩	[id]	🟥	🟥	🟥	🟥	🟥
char		🟥	🟥	[id]	🟥	🟥	🟥	🟥
int		🟩	🟩	🟩	[id]	🟥	🟥	🟥
long		🟩	🟩	🟩	🟩	[id]	🟥	🟥
float		🟩	🟩	🟩	🟩	🟩	[id]	🟥
double		🟩	🟩	🟩	🟩	🟩	🟩	[id]

Booleans can’t be converted.

Wrapper classes

Primitive wrapper class is a wrapper class that encapsulates, hides or wraps data types from the eight primitive data types

Objects of wrapper classes can be created with operator *new*.

Primitive type	Wrapper class
boolean	Boolean
byte	Byte
char	Character
float	Float
int	Integer
long	Long
short	Short
double	Double

Autoboxing and Unboxing

Autoboxing is the automatic conversion that the Java compiler makes between the primitive types and their corresponding object wrapper classes.

Converting an object of a wrapper type to its corresponding primitive value is called unboxing.

The Java compiler applies unboxing when an object of a wrapper class is:

- Passed as a parameter to a method that expects a value of the corresponding primitive type.
- Assigned to a variable of the corresponding primitive type.

References

All types except primitive types are reference types.

Value of a reference is JVM specific it can be an address of the object or something based on the address.

Value of a reference can be *null*

Default value of a reference is *null*

Size of a reference depends upon the JVM and OS.

32/64 bits in Common virtual machines for 32/64-bit systems

In Hot Spot JVM it is be 32 bits in 64-bit system if the application uses less than 32Gb of memory.

Memory allocation for new variables

Memory for a new local variable of a primitive data type is allocated in the Stack.

Memory for a new local reference is allocated in the Stack.

Memory for a new object is allocated in the Heap.

Memory for a new class member (primitive of reference) is allocated in the Heap (as a part of memory for an object).