DATA TYPES IN JAVA

www.ravindrababuravula.com

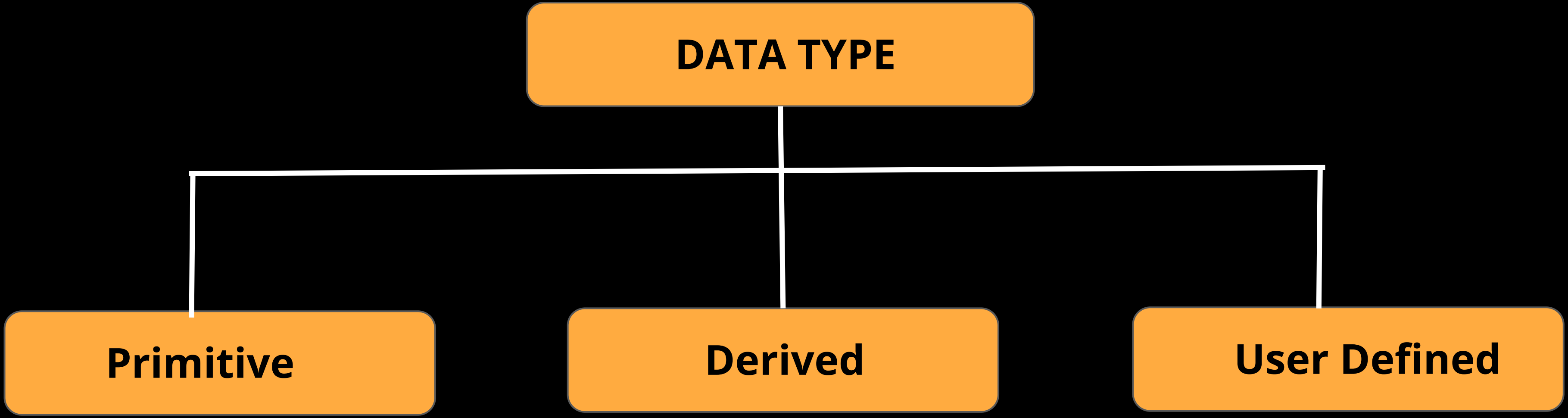
Data Types

➢ Data type is a special keyword used to allocate sufficient memory space for the data.

➢ In general every programming language is containing three categories of data types. They are

1. Fundamental or primitive data types
2. Derived data types
3. User defined data types.

Data Types



Primitive Data Types

- Primitive data types are those whose variables allows us to store only one value but they never allows us to store multiple values of same type.

Example:

```
int a;
a = 10; // valid
a = 10, 20, 30; // not valid
```

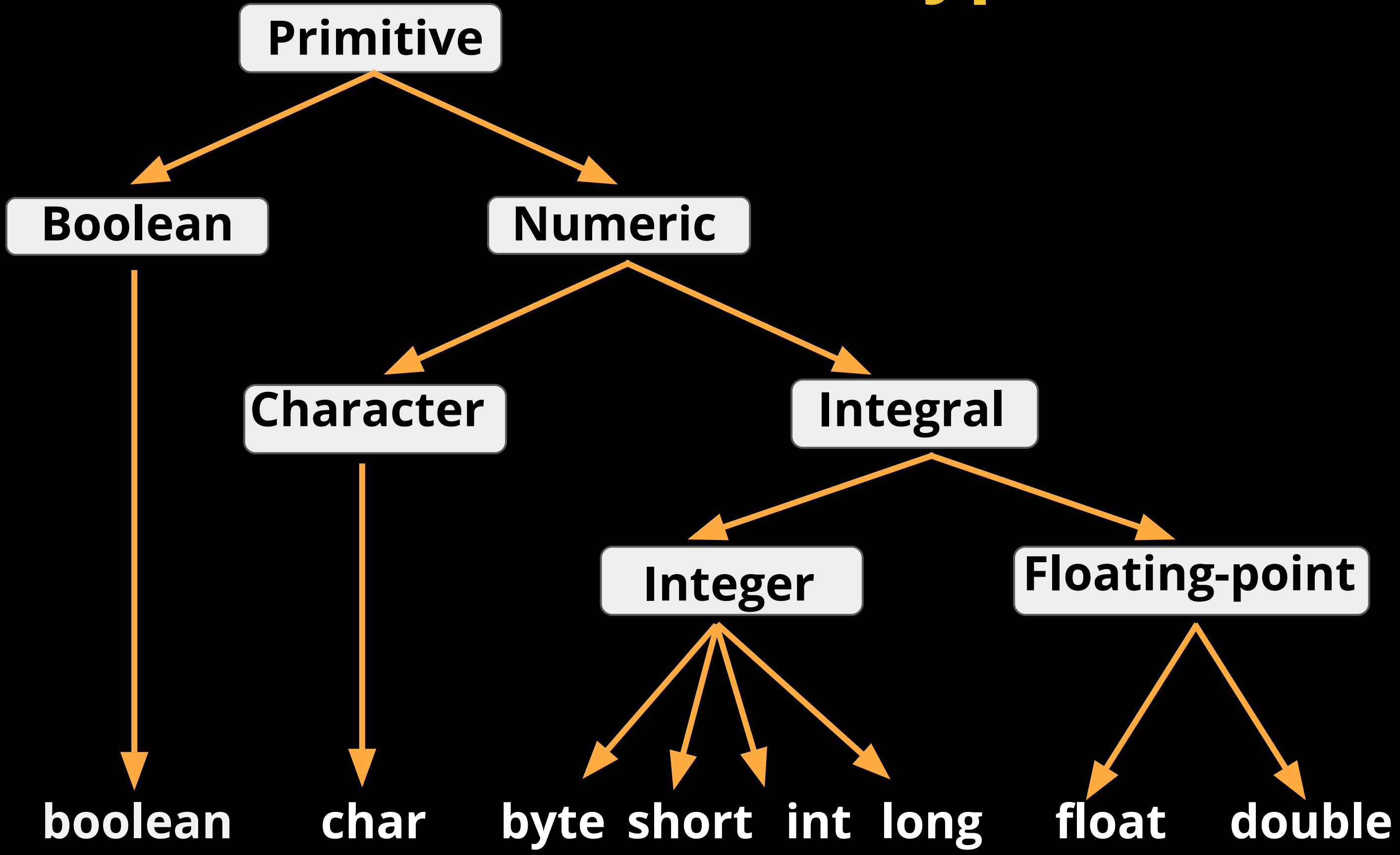
Primitive Data Types

- Java primitive data types are the basic data types that are built-in to Java language.
- A data type is a classification mechanism whereby it can be identified that what kind of data is stored inside the variable, and what operations it supports.
- Java provides a richer set of primitive or basic or built-in data types than other languages like C and C++.

Primitive Data Types

- There are eight built-in types supported by Java to support **Four Integer Type, Two Float Type, One character type, and boolean type**. All primitive or basic data types hold numeric data that is directly understood by system.

Primitive Data Types



www.ravindrababuravula.com

Integer Type

- ➢ The integer types in Java are byte, short, int, and long. These four types differ only in the number of bits and, therefore, in the range of numbers each type can represent.
- ➢ All integral types represent signed numbers. There is no unsigned keyword as there is in C and C++.

Type	Contains	Default	Size	Range
byte	Signed integer	0	8 bits	-128 to +127
short	Signed integer	0	16 bits	-32768 to +32767
int	Signed integer	0	32 bits	-2147483648 to 2147483647
long	Signed integer	0	64 bits	-9223372036854775808 to 9223372036854775807

Integer Type

Byte:

- ➢ The smallest integer type is byte.
- ➢ Byte variables are declared by use of the byte keyword.
- ➢ For example, the following declares two byte variables called a and b:

```
byte a, b;
```

Integer Type

Short:

- ➢ Short variables are declared by use of the short keyword.
- ➢ For example, the following declares two short variables called a and b:

```
short a, b;
```

Integer Type

int:

- The most commonly used integer type is int
- int variables are declared by use of the int keyword.
- For example, the following declares two int variables called a and b:

```
int a, b;
```

Integer Type

long:

- ➢ long is a signed 64-bit type and is useful for those occasions where an int type is not large enough to hold the desired value.
- ➢ long variables are declared by use of the long keyword.
- ➢ For example, the following declares two long variables called a and b:

```
long a, b;
```

Boolean Type

- ➢ The boolean type represents a truth value. There are only two possible values of this type, representing the two boolean states: on or off, yes or no, true or false.
- ➢ Java reserves the words true and false to represent these two boolean values.
- ➢ Note: boolean values can never be converted to or from other data types

Boolean Type

Type	Contains	Default	Size	Range
boolean	true or false	false	1 bit	NA

Float Type

- The float types in Java are float and double. These two types differ only in the number of bits and, therefore, in the range of numbers each type can represent.
- float is a 32-bit, single-precision floating-point value, and double is a 64-bit, double-precision floating-point value.

Float Type

Type	Contains	Default	Size	Range
float	IEEE 754 floating point	0.0	32 bits	±1.4E-45 to ±3.4028235E+38
double	IEEE 754 floating point	0.0	64 bits	±4.9E-324 to ±1.7976931348623157E+308

www.ravindrababuravula.com

Char Type

- In Java, the data type used to store characters is **char**.
- Character is 16 bits wide in Java.
- Java uses **Unicode** to represent characters.
- Java support lot of Unicode symbols from many more human languages for this purpose, it requires 16 bits.
- The range of a char is 0 to 65,536.
- There are no negative chars.

Char Type

Type	Contains	Default	Size	Range
char	Unicode character	\u0000	16 bits	\u0000 to \uFFFF

Unicode System

Unicode is a universal international standard character encoding that is capable of representing most of the world's written languages.

Why java uses Unicode System?

- Before Unicode, there were many language standards:
- ASCII (American Standard Code for Information Interchange) for the United States.
- ISO 8859-1 for Western European Language.
- KOI-8 for Russian.
- GB18030 and BIG-5 for chinese, and so on.

Problem

➢ This caused two problems:

A particular code value corresponds to different letters in the various language standards.

The encodings for languages with large character sets have variable length. Some common characters are encoded as single byte, other require two or more bytes.

Solution

- To solve these problems, a new language standard was developed i.e. Unicode System.
- In unicode, character holds 2 byte, so java also uses 2 byte for characters.
- lowest value: \u0000
- highest value: \uFFFF

Char Data type

- In addition, Java supports a number of other escape sequences that make it easy both to represent commonly used non printing ASCII characters such as newline and to escape certain punctuation characters that have special meaning in Java. For example:

```
char tab = '\t';
```

Java Escape Characters

Escape Sequence	Escape Sequence
\t	Horizontal tab
\b	Backspace
\n	New line
\f	Form feed

Java Escape Characters

Escape Sequence	Escape Sequence
\r	Carriage return
\"	Double quote
\'	Single quote
\\	Backslash

Derived Data Types

- Derived data types are those whose variables allow us to store multiple values of same type. But they never allows to store multiple values of different types.
- In general derived data type can be achieve using array.
- Example:

```
int  a[] = {10,20,30};  // valid
int b[] = {100, 'A', "ABC"};   //  invalid
```

User Defined Data types

- User defined data types are those which are developed by programmers by making use of appropriate features of the language.
- User defined data types related variables allows us to store multiple values either of same type or different type or both.
- This is a data type whose variable can hold more than one value of dissimilar type, in java it is achieved using class concept.

User Defined Data types

- ➢ Note: In java both derived and user defined data types combined name as reference data type.
- ➢ In C language, user defined data types can be developed by using struct, union, enum etc.
- ➢ In java programming user defined data type can be developed by using the features of classes and interfaces.

Type Casting

- The process of converting one data type to another is called casting.
- Casting is often necessary when a function returns a data in different form than what we need.
- Under certain circumstances Type conversion can be carried out automatically, in other cases it must be "forced" manually (explicitly)

Type Casting

- The storage size of the types while casting is very important otherwise the data loss will occur.
- For example, suppose we cast a long to an int. A long is a 64-bit value and an int is a 32-bit value. When casting a long to an int, the compiler truncates the upper 32 bits of the long value so as to fit into the 32-bit int.
- If the upper 32 bits of the long contain any useful information, then data loss will occur.

Type Casting

- In some cases, the data type of the expression is changed automatically to the variable's data type. For example, suppose that i is an integer variable declared as follows:

  int i = 10 ;

- Even though d is a variable of type double, the following assignment is valid:

  double d = i *10;   // valid, (i*10) is converted to type double

Type Casting

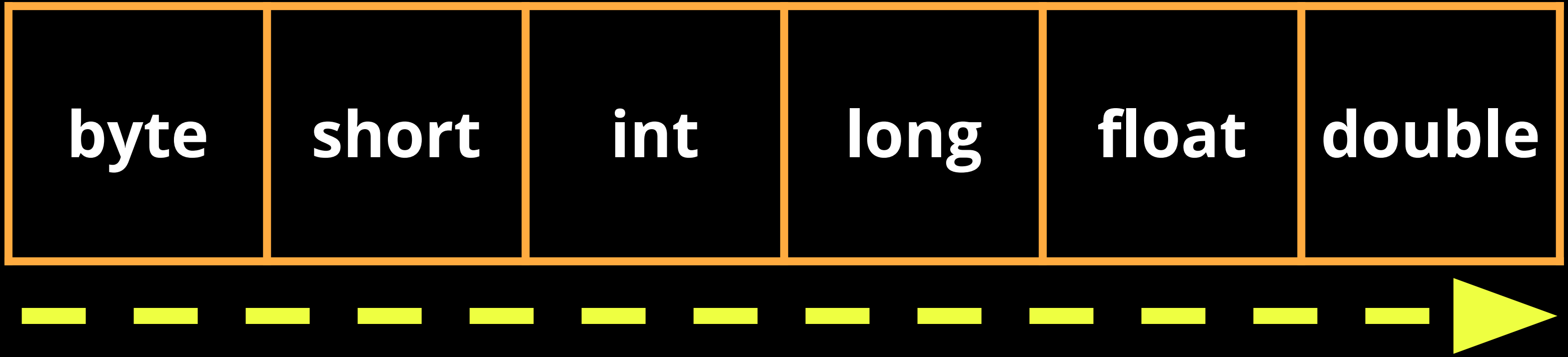
- Java changes (i*10) from int to double, and then assigns it to d.
- This conversion does not cause any loss of information because the double data type is 64 bits wide and thus is wider than the int data type that has 32 bits. This is called a widening conversion because a narrower type is converted to a wider type, and it takes place implicitly.

Type Casting

- some more examples of when widening conversions occur:
  - short s = 10;
  - int num= s;
  - float f = 10.5f;
  - double d = f;

Type Casting

➢ Following Figure summarizes the conversion rules: A conversion occurs from a given data type to any of the data types on its right.



Conversion to the wider type on the right takes place implicitly

Type Casting

- Assigning a value of type double to an int causes a compilation error:

```
double d = 100.5;
int i = d; // error!
```

- Java does not allow this assignment because a double can hold a larger range of values than an int, and it could result in a loss of data.

Type Casting

- To allow this type of assignment, it is necessary for the programmer to force a conversion from double to int using a cast.
- For example, a cast is used here to force a conversion of the variable d to type int and then assign it to i:

```
int i = (int) d; // i = 100
```

- Note: The above assignment truncates (not rounds) the fractional part so that it assigns 100 to i.

Overview of Type Casting

- **Automatic Conversion**: In Java type conversions are performed automatically when the type of the expression on the right hand side of an assignment operation can be safely promoted to the type of the variable on the left hand side of the assignment. Thus we can safely assign:

  byte -> short -> int -> long -> float -> double.

Type Casting

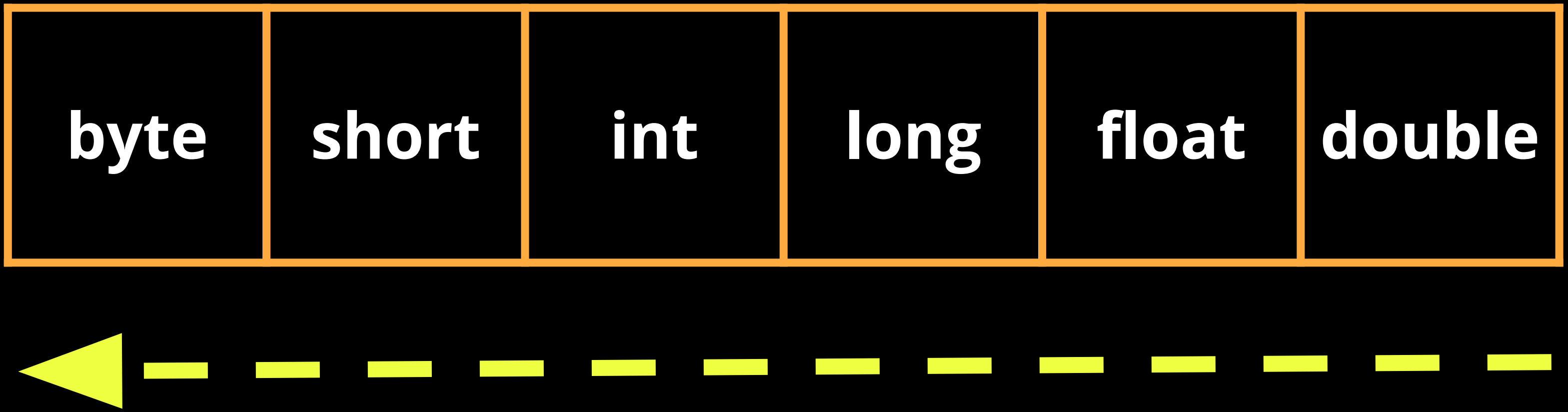
- **Explicit Conversion (Casting):** we cannot automatically convert a long to an int because the first requires more storage than the second and consequently information may be lost.
- To force such a conversion we must carry out an explicit conversion (assuming that the long integer will fit into a standard integer). This is done using a process known as a type cast

Type Casting

```
myInteger = (int) myLongInteger
```

- This tells the compiler that the type of myLongInteger must be temporarily changed to an int when the given assignment statement is processed. Thus, the cast only lasts for the duration of the assignment.

Type Casting



Conversion to the narrower type on the right takes place explicitly