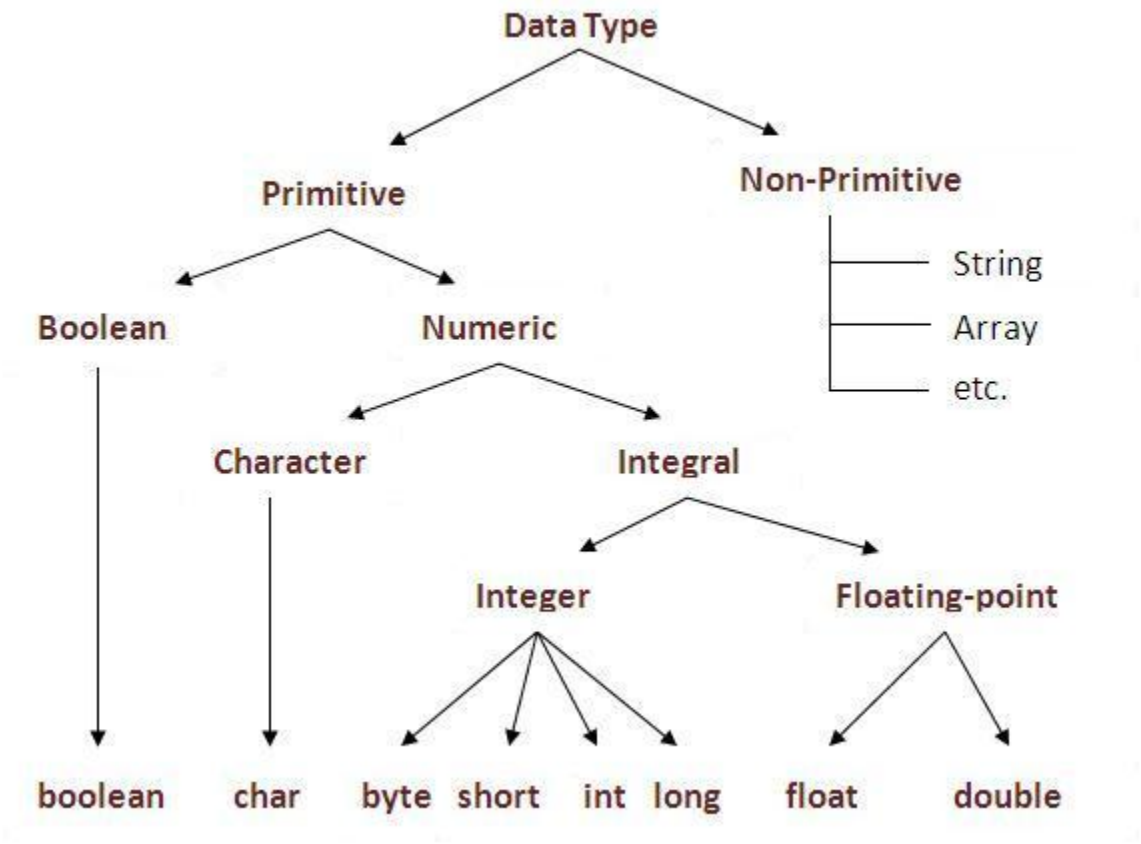
Kiểu dữ liệu trong Java

Các biến là không gì khác ngoài các vị trí bộ nhớ dành riêng để lưu các giá trị. Nghĩa là, khi bạn tạo một biến, bạn dự trữ một số không gian trong bộ nhớ.

Dựa trên kiểu dữ liệu của một biến, hệ điều hành cấp phát bộ nhớ và quyết định cái gì có thể được lưu giữ trong bộ nhớ dành riêng. Vì thế, bằng việc gán các kiểu dữ liệu khác nhau tới các biến, bạn có thể lưu giữ integer, thập phân, hoặc các ký tự trong những biến này.

Có hai kiểu dữ liệu có sẵn trong Java:

- Các kiểu dữ liệu gốc (Primitive)
- Các kiểu dữ liệu tham chiếu/đối tượng (không phải kiểu gốc Non-primitive)



Trước khi đi vào giới thiệu chi tiết về từng kiểu dữ liệu, bạn có thể theo dõi bảng tóm tắt sau:

Kiểu dữ liệu	Giá trị mặc định	Kích cỡ mặc định
boolean	false	1 bit
char	'\u0000'	2 byte
byte	0	1 byte
short	0	2 byte
int	0	4 byte
long	0L	8 byte
float	0.0f	4 byte
double	0.0d	8 byte

Câu hỏi: Tại sao char sử dụng 2 byte trong Java và \u0000 là gì?

Bởi vì Java sử dụng Unicode chứ không phải ASCII. \u0000 là dãy thấp nhất trong Unicode.

Các kiểu dữ liệu gốc trong Java

Có 8 kiểu dữ liệu gốc được hỗ trợ bởi Java. Các kiểu dữ liệu gốc này được tiền định nghĩa bởi ngôn ngữ và được định danh bởi một từ khóa. Dưới đây là chi tiết về 8 kiểu dữ liệu gốc này:

char:

- Dùng để lưu dữ liệu kiểu kí tự hoặc số nguyên không âm có kích thước 2 byte (16 bit)
- Giá trị nhỏ nhất là '\u0000' (hoặc 0) và giá trị lớn nhất là '\uffff' (hoặc 65,535)..
- Ví dụ: char nam ='IT'

Tại sao Java sử dụng Unicode?

Trước Unicode, có nhiều chuẩn ngôn ngữ như ASCII, ISO 8859-1, KOI-8, GB18030, … Điều này gây là hai vấn đề sau:

- Một giá trị code cụ thể tương ứng với các chữ cái khác nhau trong các chuẩn ngôn ngữ đa dạng.
- Mã hóa cho các ngôn ngữ với các tập ký tự lớn có độ dài biến đổi. Một số ký tự chung được mã hóa thành các byte đơn, một số khác cần hai hoặc nhiều byte.

Để xử lý các vấn đề này, một chuẩn ngôn ngữ mới được phát triển, đó là Unicode. Trong Unicode, ký tự giữ 2 byte, vì thế Java cũng sử dụng 2 byte cho các ký tự.

Kiểu dữ liệu byte:

- Dùng để lưu dữ liệu kiểu số nguyên có kích thước một byte (8 bit)
- Giá trị nhỏ nhất là -128 (-2^7) và giá trị lớn nhất là 127. (2^7 -1)
- Giá trị mặc định là 0
- Kiểu dữ liệu byte được sử dụng để lưu giữ khoảng trống trong các mảng lớn, chủ yếu là các số nguyên.
- Ví dụ: byte x = 20 , byte y = -10

Kiểu dữ liệu short:

- Dùng để lưu dữ liệu có kiểu số nguyên, kích cỡ 2 byte (16 bit).
- Giá trị nhỏ nhất là -32,768 (-2^15) và giá trị lớn nhất là 32,767 (2^15 -1).
- Kiểu dữ liệu short cũng có thể được sử dụng để lưu bộ nhớ như kiểu dữ liệu byte.
- Giá trị mặc định là 0.
- Ví dụ: short t = 50, short z = -10

int:

- Dùng để lưu dữ liệu có kiểu số nguyên, kích cỡ 4 byte (32 bit).

- Giá trị nhỏ nhất là - 2,147,483,648.(-2^31) và giá trị lớn nhất là 2,147,483,647 (2^31 -1)
- Nói chung, int được sử dụng như là kiểu dữ liệu mặc định cho các giá trị nguyên.
- Giá trị mặc định là 0.
- Ví dụ: int a = 5, int b = -50

long:

- Dùng để lưu dữ liệu có kiểu số nguyên có kích thước lên đến 8 byte.
- Giá trị nhỏ nhất là -9,223,372,036,854,775,808.(-2^63) và lớn nhất là 9,223,372,036,854,775,807. (2^63 -1)
- Kiểu này được sử dụng khi cần một dải giá trị rộng hơn int.
- Giá trị mặc định là 0L.
- Ví dụ: long a = 100000L, int b = -200000L

float:

- Dùng để lưu dữ liệu có kiểu số thực, kích cỡ 4 byte (32 bit)
- Kiểu Float được sử dụng chủ yếu để lưu bộ nhớ trong các mảng rộng hơn các số thực dấu chấm động.
- Giá trị mặc định là 0.0f.
- Kiểu Float không bao giờ được sử dụng cho các giá trị chính xác như currency.
- Ví dụ: float usd = 22.5f

double:

- Kiểu dữ liệu double được sử dụng để lưu dữ liệu có kiểu số thực có kích thước lên đến 8 byte
- Nói chung, kiểu dữ liệu này được sử dụng như là kiểu mặc định cho các giá trị decimal.
- Kiểu double không bao giờ được sử dụng cho các giá trị chính xác như currency.

- Giá trị mặc định là 0.0d.
- Ví dụ: double ct = 676.7

boolean:

- Độ lớn chỉ có 1 bit
- Dùng để lưu dữ liệu chỉ có hai trạng thái true hoặc false
- Giá trị mặc định là false.
- Ví dụ: boolean switch1 = true

Kiểu dữ liệu Object trong Java

- Các biến đối tượng được tạo bởi sử dụng các constructor đã được định nghĩa của các lớp. Chúng được sử dụng để truy cập các đối tượng. Những biến này được khai báo ở kiểu cụ thể mà không thể thay đổi. Ví dụ: Employee, Puppy, …
- Giá trị mặc định của bất kỳ biến đối tượng nào là null
- Một biến đối tượng có thể được sử dụng để tham chiếu tới bất kỳ đối tượng nào trong kiểu được khai báo hoặc bất kỳ kiểu tương thích nào.
- Ví dụ: Animal animal = new Animal("giraffe");

Literal trong Java

Một Literal hay một hằng là một code nguồn biểu diễn một giá trị cố định. Chúng được biểu diễn một cách trực tiếp trong code mà không cần kỹ thuật tính toán nào.

Các literal có thể được gán tới bất kỳ kiểu biến gốc nào. Ví dụ:

```
byte a = 12;
char a = 'CAT'
```

byte, int, long, và short có thể được biểu diễn trong hệ thập phân (cơ số 10), hệ thập lục phân (cơ số 16), hoặc hệ bát phân (cơ số 8).

Tiền tố 0 được sử dụng để chỉ hệ bát phân, và tiền tố 0x chỉ dẫn hệ cơ số 16 khi sử dụng những hệ cơ số này cho các literal. Ví dụ:

```
int decimal = 100;
int octal = 0144;
int hexa =  0x64;
```

String literal (hằng chuỗi) trong Java được xác định giống như trong hầu hết các ngôn ngữ khác bằng việc bao quanh một chuỗi ký tự liên tục trong trích dẫn kép. Ví dụ về string literal:

```
"Hello World"
"Hai\ndong"
"\"Day la trong trich dan\""
```

Các hằng chuỗi và hằng ký tự có thể chứa bất kỳ ký tự Unicode nào. Ví dụ:

```
char x = '\u0001';
String y = "\u0001";
```

Ngôn ngữ Java hỗ trợ một số dãy thoát đặc biệt cho hằng chuỗi và hằng ký tự như sau:

Ký hiệu	Biểu diễn ký tự
\n	Newline (0x0a)
\r	Carriage return (0x0d)
\f	Formfeed (0x0c)
\b	Backspace (0x08)
\s	Space (0x20)
\t	tab
\"	Trích dẫn kép

\'	Trích dẫn đơn
\\	backslash
\ddd	Octal character (ddd)
\uxxxx	Hexadecimal UNICODE character (xxxx)