Character trong Java

Thường thì, khi làm việc với các ký tự, chúng ta sử dụng kiểu dữ liệu char gốc.

Ví dụ:

```
char ch = 'a';


// Unicode for uppercase Greek omega character
char uniChar = '\u039A';


// an array of chars
char[] charArray ={ 'a', 'b', 'c', 'd', 'e' };
```

Tuy nhiên, khi mở rộng làm việc, chúng ta thường gặp các tính huống cần sử dụng các đối tượng thay vì các kiểu dữ liệu gốc. Để thực hiện điều này, Java cung cấp lớp wrapper là**Charater** với kiểu dữ liệu char gốc.

Lớp Character có một số phương thức hữu ích (ví dụ: static) để thao tác với các ký tự. Bạn có thể tạo một đối tượng Character với Character constructor.

```
Character ch = new Character('a');
```

Bộ biên dịch Java sẽ cũng tạo một đối tượng Character cho bạn. Ví dụ, nếu bạn truyền một dữ liệu char gốc vào trong một phương thức mà cho một đối tượng, bộ biên dịch tự động biến đổi kiểu char tới một Character cho bạn. Tính năng này được gọi là autobox hoặc unbox, nếu sự biến đổi này theo hướng ngược lại.

Ví dụ:

```
// Here following primitive char 'a'
// is boxed into the Character object ch
Character ch = 'a';


// Here primitive 'x' is boxed for method test,
// return is unboxed to char 'c'
char c = test('x');
```

Các ký tự ngắt văn bản trong Java

Một ký tự được đặt trước bởi một dấu gạch chéo ngược (\) là một ký tự ngắt và có ý nghĩa đặc biệt với bộ biên dịch.

Ký tự dòng mới (\n) thường được sử dụng trong bài học có trong lệnh System.out.println() để có được dòng tiếp theo sau khi chuỗi được in.

Bảng dưới liệt kê các ký tự ngắt trong Java:

Ký tự ngắt	Miêu tả
\t	Chèn một tab vào văn bản tại điểm này
\b	Chèn một backspace vào văn bản tại điểm này
\n	Chèn một dòng mới vào văn bản tại điểm này
\r	Chèn một carriage return vào văn bản tại điểm này
\f	Chèn một form feed vào văn bản tại điểm này
\'	Chèn một dấu trích dẫn đơn vào văn bản tại điểm này
\"	Chèn một dấu trích dẫn kép vào văn bản tại điểm này
\\	Chèn một ký tự dấu chéo ngược vào văn bản tại điểm này

Khi gặp một ký tự ngắt trong một lệnh print, bộ biên dịch thông dịch nó cho phù hợp.

Ví dụ:

Nếu bạn muốn đặt các trích dẫn bên trong các trích dẫn, bạn phải sử dụng ký tự ngắt, \", trong trích dẫn bên trong.

```
public class Test {


   public static void main(String args[]) {
      System.out.println("She said \"Hello!\" to me.");
   }
}
```

Nó sẽ cho kết quả:

```
She said "Hello!" to me.
```

Các phương thức của lớp Character trong Java

Dưới đây liệt kê các phương thức quan trọng mà tất cả lớp phụ của lớp Character trong Java thực thi:

STT	Phương thức và Miêu tả
1	**isLetter() trong Java** Kiểm tra có hay không giá trị char đã cho là một chữ cái
2	**isDigit() trong Java** Xác định có hay không giá trị char đã cho là một digit
3	**isWhitespace() trong Java** Xác định có hay không giá trị char đã cho là một khoảng trắng
4	**isUpperCase() trong Java** Xác định có hay không giá trị char đã cho là chữ hoa
5	**isLowerCase() trong Java**

	Xác định có hay không giá trị char đã cho là chữ thường
6	**toUpperCase() trong Java** Trả về form dạng chữ hoa của giá trị char đã cho
7	**toLowerCase() trong Java** Trả về form dạng chữ thường của giá trị char đã cho
8	**toString() trong Java** Trả về một đối tượng String biểu diễn giá trị ký tự đã cho, mà là, một chuỗi gồm một ký tự

Để có danh sách đầy đủ các phương thức, bạn tham khảo java.lang.Character API.

Chương tiếp bàn về chủ đề gì trong Java?

Trong chương tới, chúng ta sẽ tìm hiểu lớp String trong Java. Bạn sẽ học cách khai báo và sử dụng String một cách hiệu quả cũng như một số phương thức quan trọng của lớp String trong Java?