Cập nhật tháng 8 năm 2024

[Bài đọc] Các phương thức làm việc với String

Chuỗi (String) trong Java là một đối tượng của lớp String và có nhiều phương thức để làm việc với chuỗi. Dưới đây là một số phương thức cơ bản trong lớp String mà bạn sẽ thường xuyên sử dụng trong lập trình Java

1. **Length():**
   - **Mô tả:** Phương thức length() trả về số lượng ký tự trong chuỗi.
   - **Cú pháp:** int length()
   - **Giải thích:** Phương thức này trả về một số nguyên đại diện cho độ dài của chuỗi.

Ví dụ:

```
String str = "Hello, Java!";
int length = str.length();
System.out.println("Length of the string: " + length);  // Kết quả: 12
```

2. **charAt(int index):**
   - **Mô tả:** Phương thức charAt() trả về ký tự tại vị trí chỉ định trong chuỗi (chỉ số bắt đầu từ 0).
   - **Cú pháp:** char charAt(int index)
   - **Giải thích:** Phương thức này trả về ký tự nằm ở vị trí chỉ định index.

Ví dụ:

```
String str = "Hello";
char ch = str.charAt(1);
System.out.println("Character at index 1: " + ch);  // Kết quả: 'e'
```

3. **substring(int start, int end):**
   - **Mô tả:** Phương thức substring() trả về một chuỗi con (substring) bắt đầu từ chỉ số start và kết thúc trước chỉ số end.
   - **Cú pháp:** String substring(int start, int end)
   - **Giải thích:** Phương thức này lấy một phần chuỗi bắt đầu từ chỉ số start đến chỉ số end - 1. Nếu chỉ truyền một tham số start, nó sẽ lấy phần còn lại từ start đến hết

chuỗi.

Ví dụ:

```
String str = "Hello, Java!";
String subStr = str.substring(0, 5);
System.out.println("Substring: " + subStr);  // Kết quả: "Hello"
```

4. **toLowerCase():**
   - **Mô tả:** Phương thức toLowerCase() chuyển đổi tất cả các ký tự trong chuỗi thành chữ thường.
   - **Cú pháp:** String toLowerCase()
   - **Giải thích:** Phương thức này không thay đổi chuỗi ban đầu mà trả về một chuỗi mới với tất cả các ký tự là chữ thường.

Ví dụ:

```
String str = "Hello, Java!";
String lowerStr = str.toLowerCase();
System.out.println("Lowercase string: " + lowerStr);  // Kết quả: "hello, java!"
```

5. **toUpperCase():**
   - **Mô tả:** Phương thức toUpperCase() chuyển đổi tất cả các ký tự trong chuỗi thành chữ hoa.
   - **Cú pháp:** String toUpperCase()
   - **Giải thích:** Phương thức này cũng trả về một chuỗi mới với tất cả các ký tự là chữ hoa, không thay đổi chuỗi ban đầu.

Ví dụ:

```
String str = "Hello, Java!";
String upperStr = str.toUpperCase();
System.out.println("Uppercase string: " + upperStr);  // Kết quả: "HELLO, JAVA!"
```

6. **trim():**
   - **Mô tả:** Phương thức trim() loại bỏ khoảng trắng (spaces) ở đầu và cuối chuỗi.
   - **Cú pháp:** String trim()
   - **Giải thích:** Phương thức này chỉ xóa các khoảng trắng ở hai đầu chuỗi, không ảnh hưởng đến các khoảng trắng ở giữa chuỗi.

Ví dụ:

```
String str = "   Hello, Java!   ";
String trimmedStr = str.trim();
System.out.println("Trimmed string: '" + trimmedStr + "'");  // Kết quả: "Hello, Java!"
```

7. **equals(Object obj):**
   - **Mô tả:** Phương thức equals() so sánh hai chuỗi với nhau, trả về true nếu chúng có nội dung giống nhau và false nếu không.

- **Cú pháp:** boolean equals(Object obj)
- **Giải thích:** Phương thức này so sánh chuỗi hiện tại với chuỗi được truyền vào đối số, tính toán sự tương đương dựa trên nội dung chuỗi, chứ không phải tham chiếu đối tượng.

Ví dụ:

```
String str1 = "Hello";
String str2 = "Hello";
boolean isEqual = str1.equals(str2);
System.out.println(isEqual);  // Kết quả: true
```

8. **equalsIgnoreCase(String anotherString):**

- **Mô tả:** Phương thức equalsIgnoreCase() so sánh hai chuỗi mà không phân biệt chữ hoa, chữ thường.
- **Cú pháp:** boolean equalsIgnoreCase(String anotherString)
- **Giải thích:** Phương thức này giúp so sánh chuỗi mà không xét đến sự khác biệt giữa chữ hoa và chữ thường.

Ví dụ:

```
String str1 = "Hello";
String str2 = "hello";
boolean isEqual = str1.equalsIgnoreCase(str2);
System.out.println(isEqual);  // Kết quả: true
```

9. **contains(CharSequence sequence):**

- **Mô tả:** Phương thức contains() kiểm tra xem chuỗi có chứa một chuỗi con hay không.
- **Cú pháp:** boolean contains(CharSequence sequence)
- **Giải thích:** Phương thức này trả về true nếu chuỗi hiện tại chứa chuỗi con được truyền vào, ngược lại trả về false.

Ví dụ:

```
String str = "Hello, Java!";
boolean containsWord = str.contains("Java");
System.out.println(containsWord);  // Kết quả: true
```

10. **replace(char oldChar, char newChar):**

- **Mô tả:** Phương thức replace() thay thế tất cả các ký tự oldChar trong chuỗi bằng newChar.
- **Cú pháp:** String replace(char oldChar, char newChar)
- **Giải thích:** Phương thức này trả về một chuỗi mới với các ký tự được thay thế, chuỗi ban đầu không bị thay đổi.

Ví dụ:

```
String str = "Hello, Java!";
String replacedStr = str.replace( oldChar: 'J',  newChar: 'C');
System.out.println(replacedStr);  // Kết quả: "Hello, Cava!"
```

11. **indexOf(String str):**
    - **Mô tả:** Phương thức indexOf() trả về chỉ số (vị trí) của lần xuất hiện đầu tiên của chuỗi con trong chuỗi hiện tại.
    - **Cú pháp:** int indexOf(String str)
    - **Giải thích:** Nếu chuỗi con không tìm thấy, phương thức trả về -1.

Ví dụ:

```
String str = "Hello, Java!";
int index = str.indexOf("Java");
System.out.println(index);  // Kết quả: 7
```

12. **lastIndexOf(String str):**
    - **Mô tả:** Phương thức lastIndexOf() trả về chỉ số của lần xuất hiện cuối cùng của chuỗi con trong chuỗi hiện tại.
    - **Cú pháp:** int lastIndexOf(String str)
    - **Giải thích:** Phương thức này hữu ích khi bạn muốn tìm kiếm lần xuất hiện cuối cùng của một chuỗi con trong chuỗi.

Ví dụ:

```
String str = "Hello, Java! Java is fun!";
int lastIndex = str.lastIndexOf( str: "Java");
System.out.println(lastIndex);  // Kết quả: 19
```

13. **split(String regex):**
    - **Mô tả:** Phương thức split() chia chuỗi thành một mảng các chuỗi con dựa trên biểu thức chính quy (regex).
    - **Cú pháp:** String[] split(String regex)
    - **Giải thích:** Phương thức này sẽ tách chuỗi thành các phần tử trong mảng theo dấu phân cách mà bạn chỉ định.

Ví dụ:

```java
String str = "apple,banana,orange";
String[] fruits = str.split( regex: ",");
for (String fruit : fruits) {
    System.out.println(fruit);
}
// Kết quả:
// apple
// banana
// orange
```

Tài nguyên đọc thêm: https://www.w3schools.com/java/java_ref_string.asp

Danh sách các bài học