1. **Góc giải thích**

hẳn những ai biết tới Java thì không còn xa lạ gì với việc ghép các String với nhau.

String s = "Hello";

s+= " world";

System.out.println(s +"!!!");

Đây là một kiến thức cực kì cực kì cơ bản. Tuy nhiên, nếu chúng ta tăng số lượng phép nối xâu này lên thì sẽ có hệ quả gì.

Cùng xem ví dụ này nhé:

long start = System.nanoTime();

String s = "Hello";

for (int i = 0; i < 1000; i++) {

s += " world";

}

long end = System.nanoTime();

System.out.println("Total time: "+(end-start));

// Kết quả:

// Total time: 17495917 ns

// = 17.4 ms (Milliseconds)

Bây giờ, vẫn là chương trình tương tự, mình sử sụng String Buffer

long start = System.nanoTime();

StringBuffer sb = new StringBuffer("Hello");

for (int i = 0; i < 1000; i++) {

sb.append(" world");

}

String s = sb.toString();

long end = System.nanoTime();

System.out.println("Total time: "+(end-start));

// Kết quả:

// Total time: 461198 ns

// = 0.46 ms

String Buffer nhanh hơn gấp **38 lần**.

Hiệu năng được chạy trên Mac Pro 2017, tại máy bạn có thể sẽ khác, nhưng chắc chắn rằng StringBuffer luôn nhanh hơn!

**Góc giải thích**

Có một điều ít bạn học lập trình Java để ý, đó là String là immutable. Tức nội dung trong String là không được quyền thay đổi.

Nhiều bạn lầm tưởng rằng việc nối xâu là bạn thay đổi nội dung của String, nhưng thực chất bạn đang tạo ra một đối tượng hoàn toàn mới:

String s = "A";

s+="B";

// Complier sẽ tạo ra một đối tượng mới là "AB"

// Và gán vào `s`

// Bản chất `s` bây giờ là một đối tượng mới chứ bạn không hề thay đổi nội dung ban đầu của `s`.

// Đây là những gì ở dưới Compiler sẽ làm:

StringBuffer sb = new StringBuffer("A"); // Compiler Vẫn phải xài tới StringBuffer

sb.append("B");

s = sb.toString();

Vì vậy khi nối xâu trong Java, việc bạn thực hiện nó liên tục, sẽ tương đương với việc khởi tạo liên tục và nối 2 xâu lại rồi trả về đối tượng String mới dẫn tới chi phí lớn.

StringBuffer cho phép chúng ta thao tác trên một đối tượng duy nhất và thay đổi được nội dung trong nó. Nếu ban đầu nội dung là "A", bạn muốn nối thêm "B" vào. Thì nó chỉ cần gắn chuỗi bytes của "B" vào liền kề ngay sau "A" là xong. (Vì nó có thể thay đổi, khác với String là immutable).

Tới đây bạn đã hiểu rõ vài trò của StringBuffer trong Java, vì thế hãy tận dụng nó một cách tối ưu, thay vì việc cộng các String như thông thường.