

# Pemrograman Berbasis Obyek

## String, StringBuffer, StringBuilder

Oleh Politeknik Elektronika Negeri Surabaya  
2017



Politeknik Elektronika Negeri Surabaya  
Departemen Teknik Informatika dan Komputer

1

## Konten

- String Class
- Pool of Literal String
- Perbandingan String
- Concatenating String
- String Method
- Trimming and Replacing
- StringBuffer & StringBuilder
- Immutable vs Mutable



Politeknik Elektronika Negeri Surabaya  
Departemen Teknik Informatika & Komputer

2

## Class String

- Sebagaimana object java yang lain, object String bisa dibuat dengan menggunakan operator **new** (menggunakan constructor).
- Kelas String memiliki 13 constructors.
- Contoh:

```
char[] helloArray = { 'h', 'e', 'l', 'l', 'o', '.' };
String helloString = new String(helloArray);
System.out.println(helloString);
```



## String Class

- String → Object (instance dari Kelas String)
- String → immutable (constant/tidak dapat diubah)
- String berisi array of char

String str = "abc"; ← equivalent → char[] data = {a, b, c};  
 String str = new String(data);



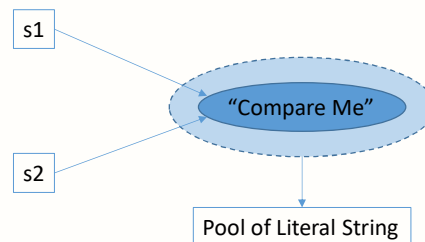
## Class String

- Java mempunyai media penyimpanan literal string yang disebut **"pool"**.
- Jika suatu literal string sudah ada di pool, Java "tidak akan membuat copy lagi".



## Pool of Literal String

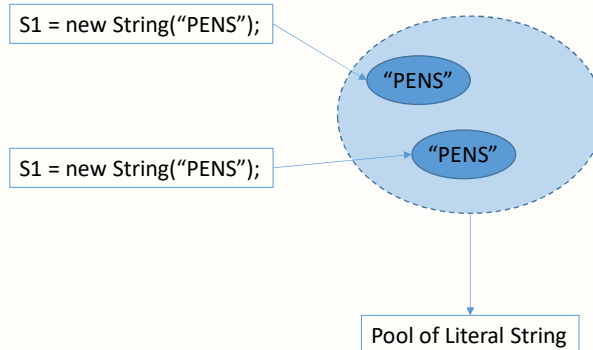
```
String s1 = "Compare me";  
String s2 = "Compare me";  
if (s1.equals(s2)) {  
    // whatever  
}  
  
if (s1 == s2) {  
    // whatever  
}
```



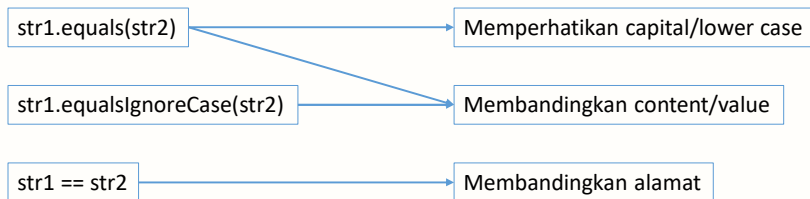
## Pool of Literal String

```
String s1 = new String("PENS");
String s2 = new String("PENS");
if (s1.equals(s2)) {
    // whatever
}

if (s1 == s2) {
    // whatever
}
```



## Perbandingan String



## Concatenating String

- `str1.concat(str2);`
- `str1 = str2 + str3;`
- `str1 += str2;`
- `str1 = str2 + int2 + double2;`

Secara otomatis diubah menjadi string



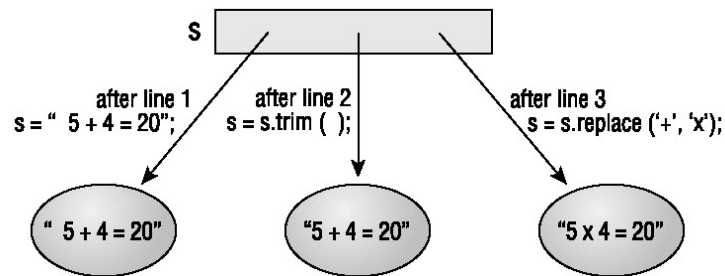
## String Method

Method	Description
<code>char charAt(int index)</code>	Mengembalikan satu karakter pada index tertentu
<code>boolean endsWith(String suffix)</code>	Mengembalikan true jika String memiliki akhiran sesuai dengan input
<code>boolean startsWith(String prefix)</code>	Mengembalikan true jika String memiliki awalan sesuai dengan input
<code>int indexOf(int ch)</code>	Mengembalikan index karakter pertama yang sesuai dengan input
<code>int lastIndexOf(int ch)</code>	Mengembalikan index karakter terakhir yang sesuai dengan input
<code>int length()</code>	Mengembalikan panjang string
<code>String substring(int startIndex)</code>	Mengembalikan String mulai dari startIndex hingga akhir String
<code>String toLowerCase()</code>	Merubah String menjadi lower case



## Trimming and Replacing

```
String s = " 5 + 4 = 20";
s = s.trim(); // "5 + 4 = 20"
s = s.replace('+', 'x'); // "5 x 4 = 20"
```



## StringBuffer dan StringBuilder

- Bersifat mutable (dapat diubah)
- Berisi rangkaian karakter (array of char) yang panjang dan isinya dapat berubah secara dinamis
- StringBuffer → synchronized
  - aman untuk digunakan dalam multiple Threads
- StringBuilder → non-synchronized
  - tidak aman digunakan dalam multiple Thread



# Immutable vs Mutable

```
String str1 = new String ("Stanford ");
str1 += "Lost!!";
```

Value berubah menjadi "Sandford Lost!"

```
StringBuffer str2 = new StringBuffer ("Stanford ");
str2.append("Lost!!");
```

Value berubah menjadi "Sandford Lost!"

- Value dari String dan StringBuffer terlihat berubah
- Menggunakan operator **+** (String) atau method **append()** (StringBuffer)
- Tapi apakah value dari String benar-benar berubah? (String bersifat Immutable)
- Mana yang lebih efisien diantara dua contoh coding tersebut?



# Immutable vs Mutable

```
String str1 = new String ("Stanford ");
str1 += "Lost!!";
```

## Generated bytecode

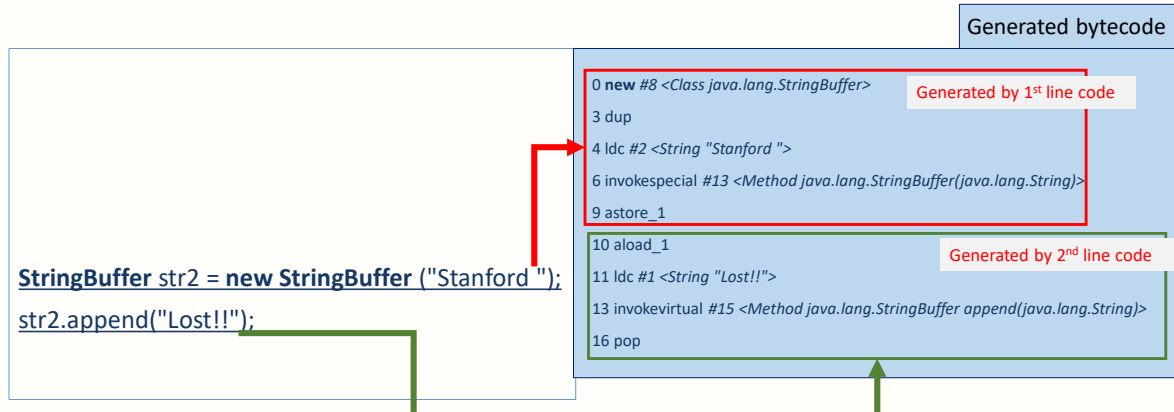
```
0 new #7 <Class java.lang.String>
3 dup
4 ldc #2 <String "Stanford ">
6 invokespecial #12 <Method java.lang.String(java.lang.String)>
9 astore_1
10 new #8 <Class java.lang.StringBuffer>
13 dup
14 aload_1
15 invokestatic #23 <Method java.lang.String valueOf(java.lang.Object)>
18 invokespecial #13 <Method java.lang.StringBuffer(java.lang.String)>
21 ldc #1 <String "Lost!!">
23 invokevirtual #15 <Method java.lang.StringBuffer append(java.lang.String)>
26 invokevirtual #22 <Method java.lang.String toString()>
29 astore_1
```

Generated by 1<sup>st</sup> line code

Generated by 2<sup>nd</sup> line code



## Immutable vs Mutable



## Tugas

1. Apakah perbedaan class String, StringBuffer dan StringBuilder?
2. Apakah yang dimaksud dengan sifat mutable dan immutable? Beri contoh!
3. Jelaskan operasi utama append dan insert yang dimiliki oleh StringBuffer!





1. Oracle Java Documentation, The Java™ Tutorials, <https://docs.oracle.com/javase/tutorial/>, Copyright © 1995, Oracle 2015.
2. Tita Karlita, Yuliana Setrowati, Rizky Yuniar Hakkun, Pemrograman Berorientasi Obyek, PENS-2012
3. Sun Java Programming, Sun Educational Services, Student Guide, Sun Microsystems, 2001.
4. John R. Hubbard, Programming With Java, McGraw-Hill, ISBN: 0-07-142040-1, 2004.
5. Patrick Niemeyer, Jonathan Knudsen, Learning Java, O'reilly, CA, ISBN: 1565927184, 2000.
6. Philip Heller, Simon Roberts, Complete Java 2 Certification Study Guide, Third Edition, Sybex, San Francisco, London, ISBN: 0-7821-4419-5, 2002.
7. Herbert Schildt, The Complete Reference, Java™ Seventh Edition, Mc Graw Hill, Osborne, ISBN: 978-0-07-163177-8, 2007