Iniciación a la programación con Java

La clase String

tilde.dis.ulpgc.es
Tecnología e Innovación para Labrar el Desarrollo Educativo

Zenón J. Hernández Figueroa

¿Qué es una String?

- Los objetos de la clase String representan secuencias de caracteres (ristras) codificadas en UTF-16.
- Se pueden crear simplemente asignando un valor literal.
- Invocando a un constructor (con la palabra new) se puede crear una ristra vacía, o una a partir de un array de caracteres, un StringBuilder, un StringBuffer u otra ristra.
- También se puede crear a partir de la conversión de un valor de otro tipo.

```
String s0 = "";
String s1 = "Hola mundo";
```

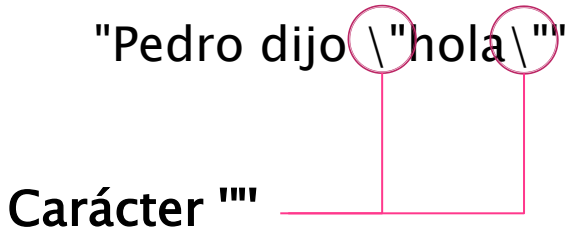
Delimitadores

```
String s2 = new String();
char[] c = {'h', 'o', 'l', 'a'};
String s3 = new String(c);
String s4 = new String(s3);
```

```
Integer i = new Integer(35);
String s5 = String.valueOf(i);
String s6 = i.toString();
```

Caracteres especiales

- Como las comillas se usan para delimitar las ristras, si una ristra debe incluir el carácter comilla, ha de usarse una "secuencia de escape" precediendo a la comilla por una barra diagonal.
- Lo mismo se hace para incluir otros caracteres especiales que no se pueden representar directamente.



Algunas secuencias de escape	
\n	Nueva línea. Coloca el cursor de la pantalla al inicio de la siguiente línea
\t	Tabulador horizontal. Desplaza el cursor hasta la siguiente posición de tab
\r	Retorno de carro. Coloca el cursor de la pantalla al inicio de la línea actual
\"	Imprime un carácter de doble comilla
\\	Imprime un carácter barra diagonal

Operaciones básicas: cálculo de la longitud y comparación

Longitud o tamaño (s.length() ==10)

```
String s = "Hola mundo"
```

```
String empty = ""
```

Ristra vacía

```
String x = "Hola mundo"
String y = "Adiós mundo"
```

Comparación

s.equals(x) == true
s.equals(y) == false
s.compareTo(x) == 0 (son iguales)
s.compareTo(y) > 0 (s es mayor que y)
y.compareTo(s) < 0 (y es menor que s)

s.equals("hola mundo") == false
s.equalsIgnoreCase("hola mundo") == true;

startsWith
endsWith
regionMatches
macthes

Operaciones básicas: concatenación de ristras

```
String s1 = "Universidad ";
String s2 = "de Las Palmas";

String s3 = s1.concat(s2);
String s4 = s3 + " de Gran Canaria";
```

s1 s2

```
s3 == "Universidad de Las Palmas";
```

```
s4 == "Universidad de Las Palmas de Gran Canaria";
```

s3

```
(s1 + (s2 + s3)).equals((s1 + s2) + s3) == true
(s1 + "").equals(s1) == true

(s1 + s2).equals(s2 + s1) == false
```

Operaciones básicas: acceso a caracteres y subristras

```
String s = "Universidad de Las Palmas de Gran Canaria"
            0123456789012345678901234567890123456789
```

```
char c = s.charAt(21) ; // c toma el valor 'l'
String s1 = s.substring(15, 25) ; // s1 toma el valor "Las Palmas"
String s1 = s.substring(29) ; // s1 toma el valor  "Gran canaria"
```

Operaciones básicas: localización de caracteres y subristras

```
String s = "Universidad de Las Palmas de Gran Canaria"
           0123456789012345678901234567890123456789
```

```
int p = s.indexOf('v') ;      // p toma el valor 3
    p = s.indexOf("de");      // p toma el valor 12
    p = s.lastIndexOf("de"); // p toma el valor 26
    p = s.indexOf('a', 24);  // p toma el valor 31
```

```
boolean b = s.contains("Las Palmas") ; // b toma el valor  true
        b = s.contains("las Palmas") ; // b toma el valor  false
```

Otras operaciones de manipulación de ristras

```
String s = "   Universidad de Las Palmas de Gran Canaria    "
```

```
String s1 = s.trim();
```

s1 = "Universidad de Las Palmas de Gran Canaria"

```
String s2 = s1.toLowerCase();
```

s2 = "universidad de las palmas de gran canaria"

```
String s3 = s1.toUpperCase();
```

s3 = "UNIVERSIDAD DE LAS PALMAS DE GRAN CANARIA"

```
String s4 = s1.replace("Universidad de", "University of");
```

s4 = "University of Las Palmas de Gran Canaria "

Tipos de ristras de caracteres

- Java ofrece tres clases para manejar ristras de caracteres:

String

Preferente

StringBuilder

Eficiencia en las modificaciones

StringBuffer

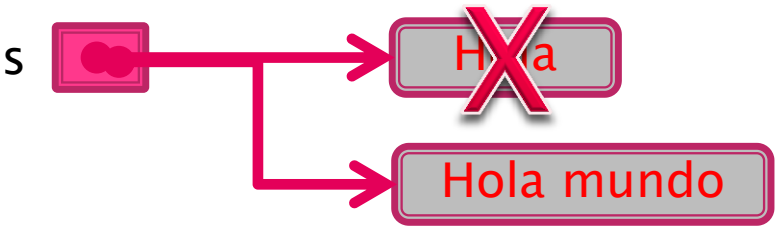
Como StringBuilder, pero seguro en ejecución concurrente

Mutable vs. inmutable

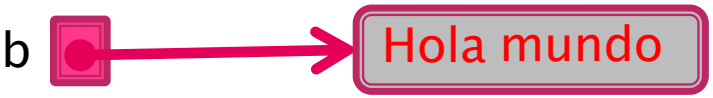
- La diferencia fundamental entre las clases String y StringBuilder es que la primera es inmutable.
- Una clase es inmutable cuando no es posible cambiar el estado de un objeto de esa clase una vez creado.

Ejemplo

```
String s = "Hola";
s = s + " mundo";
```




```
StringBuilder b = new
  StringBuilder("Hola");
b = b.append(" mundo");
```



Operaciones	Efecto