

# Linguaggi di Programmazione

a.a. 13/14

docente: Gabriele Fici

[gabriele.fici@unipa.it](mailto:gabriele.fici@unipa.it)

# 5 - Stringhe

## 5 - Stringhe

- In Java c'è la classe di sistema `String`, che permette di creare stringhe di testo
- Tuttavia, per creare una stringa non è necessario usare l'operatore `new`; basta usare un letterale stringa, cioè racchiuso tra virgolette

Esempio:

```
String a = "Pippo";  
// equivale a:  
// String a = new String("Pippo");
```

## 5 - Stringhe

- Le stringhe in Java sono in formato Unicode (estensione di ASCII)
- Ad esempio, per stampare “è” useremo il codice Unicode `\u00E9`
- Trovate i codici Unicode di tutti i simboli su [www.unicode.org/charts/](http://www.unicode.org/charts/)

Esempio:

```
String nome = "Jos\u00E9";    //stampa "Josè"
```

## 5 - Stringhe

- L'operatore `+` permette di concatenare due stringhe, oppure una stringa e un numero (che viene convertito in stringa)

Esempio:

```
String a = "Pippo";  
String nome = "Nome: " + a;
```

Esempio:

```
int j = 7;  
String indirizzo = "Interno " + j;  
                // vale "Interno 7"
```

## 5 - Stringhe

- Per leggere una stringa in input, occorre usare uno Scanner, che si occuperà di prendere i dati da console
- La classe `Scanner` è nel package `java.util`, che dovremo importare all'inizio del file

Esempio:

```
import java.util.*; //importa il package java.util
...

Scanner pippo = new Scanner(System.in);
String s = pippo.next();
```

Metterà in `s` la stringa digitata dall'utente (terminata da un carattere spaziatore o invio)

## 5 - Stringhe

- Per prendere in input un'intera riga, si usa il metodo `nextLine()`
- Per prendere in input un intero, si usa invece il metodo `nextInt()` ;
- Analogamente per gli altri tipi: `nextBoolean()`, `nextByte()`, `nextShort()`, `nextLong()`, `nextFloat()`, `nextDouble()`

Esempio:

```
Scanner dbl_in = new Scanner(System.in);  
  
double x = dbl_in.nextDouble();
```

## 5 - Stringhe

- Per avere la lunghezza di una stringa si usa il metodo `length`, che è di tipo `int` e non ha parametri

Esempio:

```
String a = "Pippo";  
  
int j = a.length();      // j vale 5
```



## 5 - Stringhe

- Il metodo `substring` restituisce una sottostringa di una stringa specificando il primo carattere (incluso) e l'ultimo carattere (escluso)
- Attenzione: come in C, il primo indice è 0
- Omettere il secondo parametro equivale a prendere tutto il suffisso

Esempio:

```
String s = "Ciao a tutti!";  
  
String t = s.substring(1,4);    // t vale "iao"  
String v = s.substring(10);    // v vale "ti!"
```

## 5 - Stringhe

- Per confrontare due stringhe si usa il metodo `equals`, che ritorna un boolean

Esempio:

```
String s = "ab", s1 = "a" + "b";  
  
boolean c = s.equals(s1);    // c vale true
```

- Per confrontare due stringhe senza tenere conto di maiuscole e minuscole si usa il metodo `equalsIgnoreCase`

## 5 - Stringhe

- I metodi `toLowerCase` e `toUpperCase` trasformano i caratteri di una stringa tutti in minuscolo e tutti in maiuscolo, rispettivamente
- Non hanno parametri e restituiscono una stringa

Esempio:

```
String s = "Ciao";
```

```
String s_low = s.toLowerCase(); //s_low è "ciao"
```

```
String s_upp = s.toUpperCase(); //s_upp è "CIAO"
```

## 5 - Stringhe

- Il metodo `charAt` permette di estrarre il carattere che si trova in una data posizione
- Ha parametro intero e restituisce un carattere

Esempio:

```
String s = "Ciao!";
```

```
System.out.println(s.charAt(4)); //stampa "!"
```

## 5 - Stringhe

- Il metodo `replaceFirst` permette di sostituire la prima occorrenza di una sequenza di caratteri con un'altra
- Ha due parametri: la sequenza da sostituire e quella con cui sostituirla; restituisce la stringa modificata

Esempio:

```
String s = "Ciao!";

String t = s.replaceFirst( "!" , "...");
// N.B. s non viene cambiata!

System.out.println(t);           //stampa "Ciao..."
```

## 5 - Stringhe

- Analogamente, il metodo `replaceAll` permette di sostituire tutte le occorrenze di una sequenza di caratteri con un'altra
- Ha due parametri: la sequenza da sostituire e quella con cui sostituirla; restituisce la stringa modificata

Esempio:

```
String s = "Ciao!!!";  
  
s = s.replaceAll( "!" , "." );  
System.out.println(s);      //stampa "Ciao..."
```

## 5 - Stringhe

- Il metodo (statico) `String.valueOf` permette di trasformare un qualsiasi tipo primitivo in una stringa

Esempio:

```
int num = 1005609;

String snum = String.valueOf(num);
snum = snum.replaceAll( "0" , "X" );
System.out.println(snum);           //stampa "1XX56X9"
```