

La classe String

Programmazione
Corso di laurea in Informatica

Una classe che usa la classe String

- Una classe con un unico metodo `diCiao()` che stampi un messaggio di saluto
 - Esempio [Saluti.java](#)
- Il metodo `diCiao()` si conclude riportando all'ambiente chiamante un valore del tipo dichiarato `String`
- con l'istruzione `return`

```
return espressione;
return;
```
- Modifichiamo la classe per aggiungere il campo d'istanza `nome`
 - Esempio [Saluti_2.java](#)

AA 2007/08 © Alberti 2 Programmazione 7. La classe String

Campi d'istanza

- Ciascun oggetto di una classe possiede una propria copia di un variabile dichiarata nella classe, come variabile d'istanza.
- Spesso dichiarata `private` per realizzare la protezione dei dati, mediante *incapsulamento*

```
Saluti_2 Andrea;
Andrea = new Saluti_2 ("Andrea");
```

AA 2007/08 © Alberti 3 Programmazione 7. La classe String

Effettuare un test della classe

- Va definito un programma che spesso si chiama *driver* che ha lo scopo di generare oggetti della classe di cui volete fare un collaudo e ne invoca i metodi
- Valutare i risultati ed eventualmente ridefinire la classe per migliorarla o correggerla
- [TestSaluti.java](#) [TestSaluti_2.java](#)

AA 2007/08 © Alberti 4 Programmazione 7. La classe String

Errore comune

- Dimenticare l'inizializzazione di variabili oggetto

```
Rectangle mio_rettangolo;
* mio_rettangolo.translate (5, 5);
Saluti_2 salutaCarlo;
* salutaCarlo.diCiao();
```
- Le istruzioni `*` generano un *errore*: si applica un metodo a un oggetto che non esiste ancora
- La dichiarazione serve solo per creare la variabile oggetto sullo *stack*, ma non per inicializzarla;
- L'inizializzazione genera un oggetto sullo *heap* va effettuata esplicitamente mediante la chiamata all'operatore `new`

AA 2007/08 © Alberti 5 Programmazione 7. La classe String

Costruttori vs metodi

- I costruttori non sono metodi
- I costruttori non possono essere invocati su oggetti esistenti
- I costruttori non vengono invocati come i metodi mediante l'operatore `dot` (`.`)
- I costruttori vengono invocati solo all'atto della generazione di un oggetto tramite l'operatore `new`
- Errore:

```
Saluti_2 persona;
* persona.Saluti_2 ("Andrea");
```

AA 2007/08 © Alberti 6 Programmazione 7. La classe String

Concatenazione di stringhe

- L'operatore di concatenazione di stringhe **+** viene usato per appendere una stringa ad un'altra
- Il simbolo **+** è anche usato per l'operazione di addizione aritmetica
 - La funzione che viene eseguita dall'operatore **+** dipende dal tipo di informazione su cui opera
 - Se entrambi gli operandi sono stringhe, o una è una stringa e l'altra è un numero, esegue la *concatenazione di stringhe*
 - Se entrambi gli operandi sono numeri, allora li *somma*
- L'operatore **+** viene valutato da sinistra a destra
- Le parentesi possono essere usate per alterare l'ordine di esecuzione
- Esempio [Addition.java](#)

AA 2007/08
© Alberti

7

Programmazione
7. La classe String

Sequenze di escape

- Alcuni caratteri speciali possono dare luogo ad ambiguità
- Il compilatore interpreta la seconda occorrenza di **"** come la fine della stringa letterale
- Una *sequenza di escape* è un insieme di caratteri che ne rappresenta uno speciale
- La sequenza di escape inizia con il *carattere backslash* **** e indica che il carattere che segue va trattato in modo speciale

```
System.out.println ("Ti ho detto \"Ciao!\");
```

AA 2007/08
© Alberti

8

Programmazione
7. La classe String

Sequenze di escape

- Alcune sequenze di escape in Java:

Sequenza	Significato
<code>\b</code>	backspace
<code>\t</code>	tabulazione
<code>\n</code>	newline
<code>\r</code>	ritorno
<code>\"</code>	virgolette doppie
<code>'</code>	virgoletta
<code>\\</code>	backslash

- esempio [Roses.java](#)

AA 2007/08
© Alberti

9

Programmazione
7. La classe String

La classe predefinita String

- Java mette a disposizione la classe predefinita **String** per rappresentare stringhe di caratteri
- Ogni stringa letterale, delimitata dai segni **"**, **"** è un oggetto della classe **String**
 - Una stringa letterale non può essere spezzata su più righe nel codice
- Esempio [Fatti.java](#)

AA 2007/08
© Alberti

10

Programmazione
7. La classe String

Gli oggetti stringhe

- Per *istanziare* l'oggetto stringa **"Corso di Prog ..."**
- Un oggetto è un'istanza di una particolare classe

```
titolo = new String ("Corso di Programmazione");
```

operatore di assegnamento

variabile

chiamata al costruttore genera un riferimento da assegnare

AA 2007/08
© Alberti

11

Programmazione
7. La classe String

Creare oggetti stringhe

- Per la sola classe **String** non è necessario invocare il costruttore **new** per creare un oggetto stringa
- Speciale sintassi che vale solo per questa classe
- Il riferimento all'oggetto di tipo **String** creato implicitamente viene passato come parametro al metodo **println**

```
corso = "Programmazione";  
corso = new String ("Programmazione");
```

```
System.out.println ("Prima le cose importanti");
```

AA 2007/08
© Alberti

12

Programmazione
7. La classe String

La classe String

- La classe **String** ha diversi metodi per manipolare stringhe
- Dato un oggetto della classe, possiamo usare l'operatore **dot** per invocare i metodi
`corso.length()`
- Molti metodi *riportano un valore* mediante l'istruzione **return**
 - Come un intero o un nuovo oggetto di tipo **String**
- Esempio [StringMutation.java](#)

AA 2007/08 © Alberti 13 Programmazione 7. La classe String

Metodi per stringhe

Tipo di ritorno	Nome del metodo	Argomenti
String	toUpperCase	
char	charAt	int indice
String	concat	String con
int	length	
String	substring	int da, int a
int	compareTo	String conStr
String	replace	char v, char n
boolean	startsWith	String prefisso

AA 2007/08 © Alberti 14 Programmazione 7. La classe String

Prototipi e signature

- La **segnatura** o **firma** di un metodo è costituita dal
 - Nome del metodo
 - I tipi dei parametri
- Per utilizzare un metodo occorre conoscerne anche il **prototipo**
 - Segnatura
 - Tipo del valore di ritorno
 - Se il metodo non restituisce nulla si dichiara **void** il tipo della restituzione
- Overloading di metodi**, stesso nome diversa segnatura
 - Il compilatore sceglie il metodo in base agli argomenti usati
 - Come il metodo **+** usato per sommare e concatenare

AA 2007/08 © Alberti 15 Programmazione 7. La classe String

Esempi di prototipi

```
int compareTo(String)
boolean equals(String str)
int length()
String toUpperCase()
String substring(int, int)
```

AA 2007/08 © Alberti 16 Programmazione 7. La classe String

Metodo toLowerCase

- Restituisce il riferimento a una nuova stringa costituita dagli stessi caratteri della stringa che esegue il metodo, con le eventuali lettere maiuscole trasformate in minuscole

Tipo restituito	Nome del metodo	Argomenti
String	toLowerCase	nessuno

```
String s1 = "CIAO";
String s2 = s1.toLowerCase(); equivalente a:
String s2 = "CIAO".toLowerCase();
```

AA 2007/08 © Alberti 17 Programmazione 7. La classe String

Nota bene ...

```
String parola = "CIAO";
String nuovaParola = parola.toLowerCase();
```

AA 2007/08 © Alberti 18 Programmazione 7. La classe String

Metodo length

- Restituisce un `int`, cioè un numero intero, uguale alla lunghezza della stringa rappresentata dall'oggetto che esegue il metodo.

Tipo restituito	Nome del metodo	Argomenti
<code>int</code>	<code>length</code>	

`"ciao".length()` restituisce la stringa `4`

Metodo concat

- Restituisce un riferimento alla stringa ottenuta concatenando alla stringa che esegue il metodo la stringa fornita come argomento.

Tipo restituito	Nome del metodo	Argomenti
<code>String</code>	<code>concat</code>	<code>String</code>

`String risposta = "Ciao ".concat(nome).concat("!");`

`String risposta = "Ciao ".concat(nome.concat("!"));`

Metodo substring

- Restituisce il riferimento a una stringa formata dai caratteri che vanno dalla posizione `x` fino alla posizione `y - 1` della stringa che esegue il metodo.

Tipo restituito	Nome del metodo	Argomenti
<code>String</code>	<code>substring</code>	<code>int, int</code>

`"distuggere"` `d` è in posizione 0, l'ultima `e` in posizione 10

`"distuggere".substring(2, 9)` restituisce la stringa `"strugge"`

Metodo substring

- Restituisce un riferimento a una stringa formata da tutti i caratteri della stringa, che esegue il metodo, che si trovano tra la posizione specificata nell'unico argomento e la fine della stringa.

Tipo restituito	Nome del metodo	Argomenti
<code>String</code>	<code>substring</code>	<code>int</code>

`"distuggere".substring(8)` restituisce la stringa `"ere"`