Corso di Laurea Ingegneria Informatica

Fondamenti di Informatica 1

Dispensa 3

Programmazione Java

Alfonso Miola

Settembre 2007

Contenuti

- Il linguaggio Java
- Applicazioni Java e il metodo main
- Esempi di applicazioni
  - Scrittore sullo schermo
  - Radice quadrata
- Esempi di classi e oggetti Java
  - Gli oggetti String, i loro metodi e le possibili applicazioni

Prerequisiti

Questo capitolo presuppone la conoscenza degli argomenti già trattati nelle precedenti lezioni di questo corso e del corso di Laboratorio di Informatica, con particolare riferimento al capitolo 1 e al capitolo 2 del libro di testo

Il linguaggio di programmazione Java

Java è

- un linguaggio di programmazione
  - un **linguaggio di programmazione orientato agli oggetti**
- indipendente dalla piattaforma
  - realizzato per tipi diversi di calcolatori
    - *write once, run everywhere*
- realizzato per le reti di calcolatori
  - supportato dai principali browser Web
  - sviluppato dalla Sun Microsystems, e rilasciato nel 1995

Java è un linguaggio di programmazione

- **Ogni linguaggio di programmazione, e quindi anche Java, è caratterizzato da:**
  - **la sintassi - l'insieme delle regole grammaticali per scrivere i programmi con quel linguaggio**
  - **la semantica - l'insieme delle regole che stabiliscono il significato dei programmi, ovvero il modo in cui i programmi devono essere interpretati ed eseguiti dal calcolatore**

La successiva dispensa tratterà questi argomenti in dettaglio

Java è orientato agli oggetti . . .

- **Nel paradigma di programmazione orientato agli oggetti**
  - **l'esecuzione di un programma consiste nella cooperazione di un insieme di oggetti**
  - **un programma è la descrizione di un insieme di oggetti**

- **Il progetto (la descrizione) di un oggetto è definito da una classe**

. . . Java è orientato agli oggetti

- **In Java, l'unità fondamentale di programmazione è la classe**
  - **Java permette di definire classi**
  - **una classe può essere il progetto di un singolo oggetto (un oggetto classe) o di oggetti (oggetti istanza) di una certa tipologia**
  - **una classe implementa il comportamento e lo stato di una certa tipologia di oggetti**
    - **le operazioni sono implementate da metodi**
    - **le proprietà sono rappresentate da variabili**

Programmi Java

- **In Java, ogni programma è un oggetto: esistono due tipi di programmi Java**
  - **applicazioni Java**
    - un programma a sé stante, che viene eseguito da un interprete Java, chiamato macchina virtuale Java (o JVM, Java Virtual Machine)
  - **applet Java**
    - un programma immerso in una pagina web, che viene eseguito dalla JVM di un browser web
- **Un programma Java consiste nella definizione di un insieme di classi**
  - una classe che modella il programma
  - una classe per ciascuna tipologia di oggetti necessari per l'esecuzione del programma

Applicazioni Java

- **Una applicazione è una classe applicazione**
  - **che definisce un oggetto classe e che sa eseguire esclusivamente l'operazione speciale main**
- **In una applicazione Java**
  - **l'applicazione è rappresentata da un oggetto classe**
  - **l'utente dell'applicazione può interagire solo con alcuni degli oggetti coinvolti dall'applicazione**
  - **l'utente dell'applicazione può sicuramente interagire con l'oggetto classe che rappresenta l'applicazione**
  - **l'utente dell'applicazione può richiedere all'oggetto classe che rappresenta l'applicazione solo di eseguire l'operazione speciale main**

API di Java . . .

- **L'ambiente di sviluppo per Java è corredato da un numeroso insieme di classi e oggetti predefiniti - le API (Application Programming Interface) di Java**
  - **le API sono in insieme di elementi standard di programmazione per la gestione di aspetti applicativi di carattere generale — input-output, stringhe, grafica, ...**
    - **le classi delle API sono raggruppate in "librerie" — chiamate package**
    - **le API rappresentano quindi un supporto che permettono al programmatore di concentrarsi sullo sviluppo di classi e oggetti relativi alle applicazioni di interesse**

. . . API di Java

- Le API rappresentano quindi un supporto al programmatore che gli permettono di concentrarsi sullo sviluppo di classi e oggetti relativi alle applicazioni di interesse
- Gli oggetti e le classi delle API di Java possono essere utilizzati conoscendone l'interfaccia
  - l'interfaccia degli elementi delle API di Java è documentata da un insieme di pagine web disponibile al programmatore

Programmazione in Java . . .

- **La programmazione in Java coinvolge i seguenti aspetti**
  - **conoscenza del linguaggio Java**
    - **ovvero, conoscenza della sintassi e della semantica di Java**
  - **uso di oggetti e classi predefiniti (ad esempio, definiti nelle API di Java o in altri package a disposizione) di cui si conoscano le interfacce**
  - **definizione di nuove classi Java**
    - **per implementare programmi, oggetti classe o tipologie di oggetti istanza necessari alla soluzione del problema**

. . . Programmazione in Java

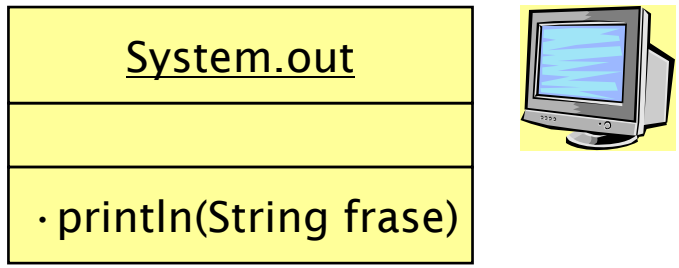
- Viene ora mostrata una panoramica sulla programmazione in Java
- Come nell'apprendimento dei linguaggi naturali anche qui si incomincia imparando a "leggere"
  - viene mostrato come leggere alcuni programmi Java, di complessità via via crescente
- Solo successivamente impareremo a "scrivere" e quindi a . . . "parlare Java"
  - alla progettazione e scrittura di programmi Java sono dedicati molti dei successivi capitoli

Scrittore sullo schermo

- Si vuole scrivere una applicazione Java che visualizza sullo schermo le seguenti frasi
  - Questo corso
  - introduce i concetti di
  - base dell'informatica
- Più precisamente, si vuole definire una classe che è il progetto di un oggetto classe che
  - è una classe applicazione
  - sa eseguire una operazione il cui effetto è quello di visualizzare queste tre frasi sullo schermo

System.out

- **L'applicazione ha bisogno di visualizzare una frase sullo schermo**
  - **l'applicazione può utilizzare l'oggetto System.out**

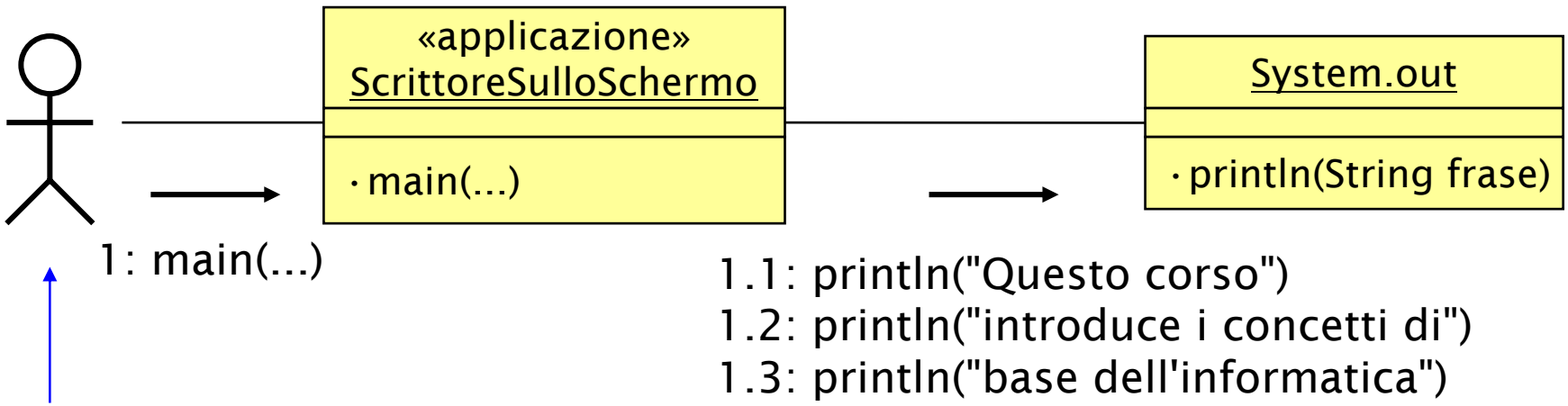


  - **System.out è un oggetto definito dalle API di Java**
  - **System.out modella lo schermo del calcolatore**
  - **System.out sa eseguire una operazione println che visualizza una frase (che è il parametro dell'operazione)**

L’applicazione ScrittoreSulloSchermo

```
/* Applicazione che visualizza una frase sullo
  schermo. */
class ScrittoreSulloSchermo {
  public static void main(String[] args) {
    System.out.println("Questo corso");
    System.out.println("introduce i concetti di");
    System.out.println("base dell'informatica");
  }
}
```

Interazione fra oggetti



questa icona rappresenta l'utente del programma

- **Questo "Diagramma di collaborazione", nel linguaggio UML, descrive i messaggi scambiati quando un utente richiede al calcolatore di eseguire l'applicazione ScrittoreSulloSchermo**

Esercizio

- **Commentare la definizione della seguente classe**

```
/* Applicazione che visualizza sullo schermo
 * la poesia Mattino di Giuseppe Ungaretti. */
class Mattino {
    public static void main(String[] args) {
        System.out.println("M'illumino");
        System.out.println("d'immenso");
    }
}
```

Esercizio

- Ora che abbiamo imparato a leggere proviamo anche a “scrivere” in Java !
- Una lunga tradizione vuole che il primo programma scritto da un programmatore sia quello che visualizza sullo schermo la frase Hello, world (che significa “Ciao, mondo”)
  - definire l’applicazione Java CiaoMondo che visualizza sullo schermo la frase Hello, world

Ovviamente possiamo anche scrivere altro a piacere

Errori di programmazione

- **Durante la scrittura di classi è possibile commettere degli errori di programmazione**
  - **possibili errori di programmazione**
    - **la frase scritta non è corretta nel linguaggio di programmazione (errori grammaticali)**
    - **la frase scritta è corretta nel linguaggio di programmazione, ma il suo significato è diverso da quanto ci si era prefissi (errori non grammaticali)**
- **Va inoltre osservato che**
  - **l'individuazione degli errori grammaticali è solitamente supportata dagli strumenti di programmazione (in particolare, dal compilatore)**
  - **viceversa, l'individuazione degli errori non grammaticali non è supportata dagli strumenti di programmazione, ed è quindi necessario ricorrere a opportune metodologie**

Errori comuni

❑ Alcuni possibili errori di programmazione

- **omissione o uso errato della punteggiatura**
  - **ad esempio, omettere i punti e virgola — sono solitamente errori grammaticali, ma non sempre**
- **uso errato delle parentesi**
  - **ad esempio, dimenticare di chiudere una parentesi graffa**
- **uso errato delle lettere maiuscole e minuscole**
  - **ad esempio, scrivere Class anziché class (errore grammaticale) — oppure Main anziché main (errore non grammaticale)**
- **errori di battitura**
  - **ad esempio, scrivere viod anziché void**
- **dimenticare parole**
- **inserire spazi erronei**
  - **ad esempio, scrivere Scrittore Sullo Schermo**
- **invertire l'ordine delle parole o delle istruzioni**

Calcolo di una radice quadrata

- **Si vuole scrivere una applicazione Java che calcola e visualizza la radice quadrata di 144**
  - l'esecuzione di questo programma dovrà visualizzare sullo schermo il numero 12

Math
double sqrt(double n)

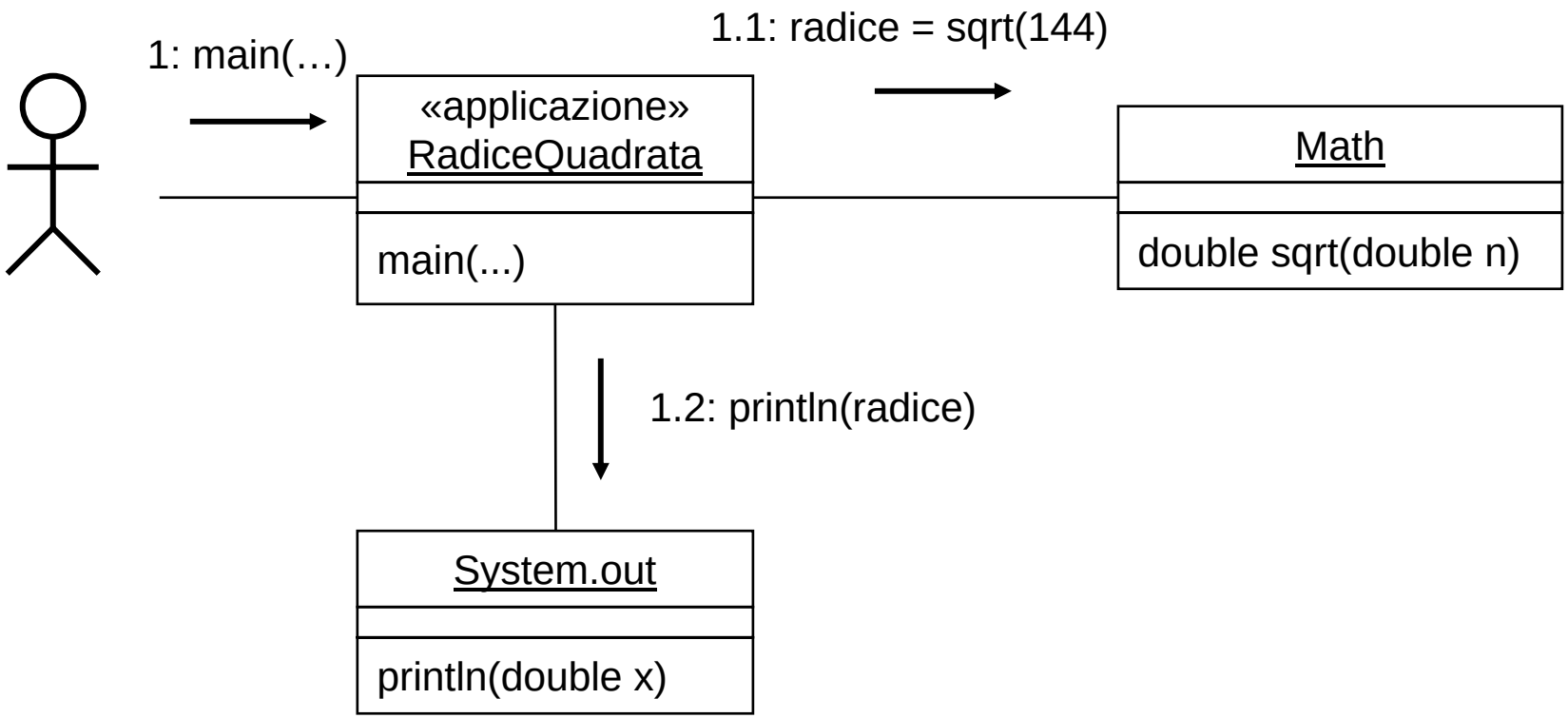
System.out
println(double x)

- **Math, come System.out e altre classi e oggetti, è un oggetto predefinito nelle API (Application Programming Interface) di Java**

L’applicazione RadiceQuadrata

```
/* Applicazione che calcola e
 * visualizza sullo schermo
 * la radice quadrata di 144. */
class RadiceQuadrata {
    public static void main(String[] args) {
        double radice;
        radice = Math.sqrt(144);
        System.out.println(radice);
    }
}
```

Diagramma di collaborazione per RadiceQuadrata

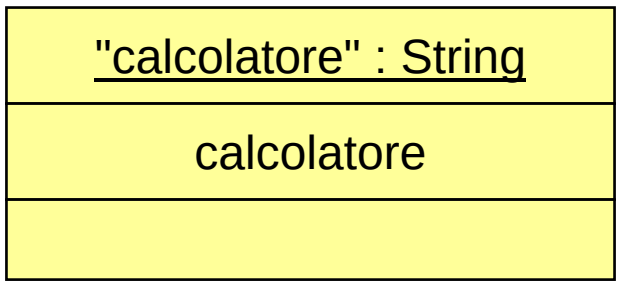


Stringhe e oggetti String

- **Una stringa è una sequenza finita di caratteri**
  - le stringhe sono un tipo di dati di uso molto comune – consentono di rappresentare informazioni testuali e descrittive
- **Java supporta la gestione di stringhe attraverso la classe String del package java.lang**
  - un oggetto String rappresenta una sequenza finita di caratteri dell'alfabeto Unicode
  - gli oggetti String sanno eseguire operazioni utili per la loro gestione facenti parte dell'interfaccia della classe

Stringhe e oggetti String

- **Il valore (o contenuto) di un oggetto String è la stringa rappresentata da quell'oggetto**
  - **ad esempio, il letterale "calcolatore" è un nome, cioè denota un oggetto String, il cui valore (stato) è la stringa calcolatore**



Letterali stringa

- **Un letterale String è una stringa racchiusa tra doppi apici " e "**
  - **ad esempio, "calcolatore"**
  - **il valore di un letterale String è la stringa racchiusa tra apici (apici esclusi)**
  - **l'uso di un letterale String è equivalente alla creazione di un oggetto di tipo String**
  - **un letterale String è una espressione di tipo String**
  - **ad esempio, "calcolatore" è il riferimento a un oggetto String con valore la stringa calcolatore**

Uso di variabili riferimento

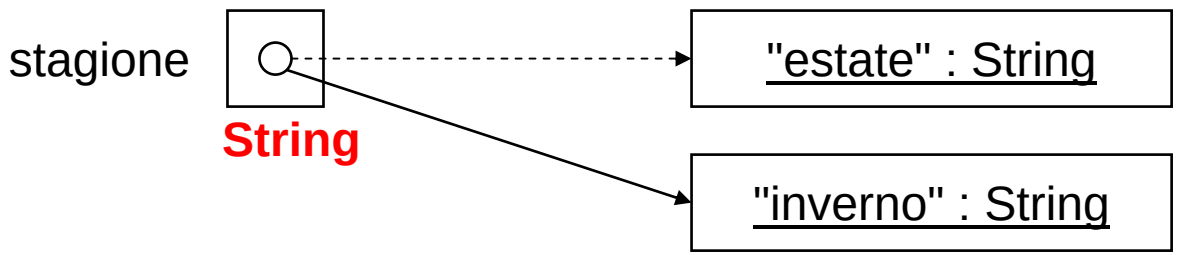
- **Essendo String il nome di una classe, è possibile dichiarare variabili di tipo String**
  - **si tratta di variabili riferimento cioè di variabili che permettono di memorizzare il riferimento a un oggetto ma non il valore dell'oggetto – che è memorizzato in un'altra area di memoria**

```
String stagione;
stagione = "estate";
System.out.println(stagione); // visualizza estate
```

Variabili riferimento e assegnazioni

- **Se a una variabile riferimento (che referenzia già un oggetto A) viene assegnato il riferimento a un altro oggetto B**
  - **il riferimento all'oggetto A memorizzato dalla variabile viene sostituito dal riferimento all'oggetto B e l'oggetto A continua ad esistere**

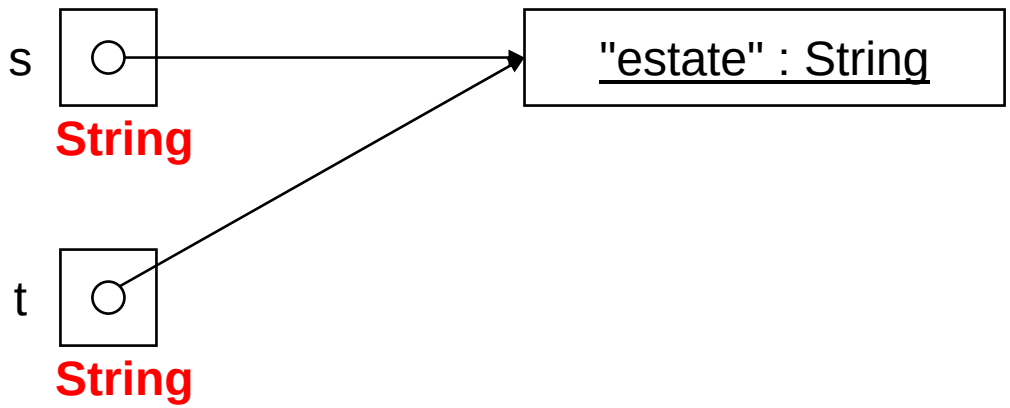
```
String stagione;
stagione = "estate";
stagione = "inverno";
```



Un oggetto può essere referenziato da più variabili

- È possibile che un oggetto sia contemporaneamente referenziato da più variabili

```
String s, t;
s = "estate";
t = s; // copia in t il riferimento memorizzato da s
```

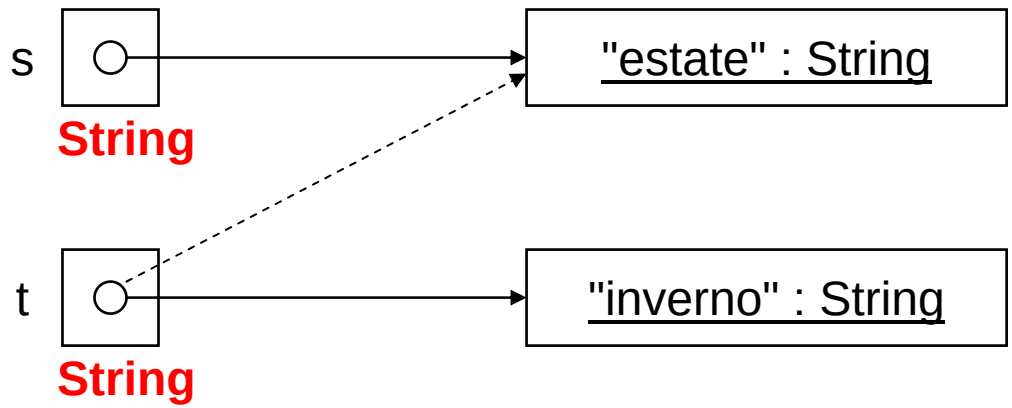


Le variabili sono indipendenti

Le variabili sono indipendenti

- una assegnazione a una variabile modifica il valore di una sola variabile

```
String s, t;
s = "estate";
t = s; // copia in t il riferimento memorizzato da s
t = "inverno";
```



Gli oggetti String sono immutabili

- **Un oggetto String rappresenta una stringa costante, immutabile**
  - **dopo la creazione, il valore di un oggetto String non può essere modificato**
  - **l'unico modo di interagire con un oggetto String è mediante i metodi della classe String**
  - **nessun metodo della classe String modifica l'oggetto su cui il metodo viene invocato**

Uso di oggetti String

- **Che cosa è un oggetto String?**
  - dal punto di vista del contenuto informativo, un oggetto String rappresenta una stringa
  - dal punto di vista comportamentale, un oggetto String è un oggetto che sa eseguire un certo numero di operazioni per la manipolazione di se stesso
- **La classe String è usata per rappresentare il tipo delle stringhe**
  - un insieme di elementi — le stringhe
  - un insieme di operazioni, per la manipolazione degli elementi di un tale insieme

Il metodo int length()

- **Il metodo int length() della classe String**
  - calcola la lunghezza della stringa
  - la lunghezza di una stringa è il numero di caratteri della sequenza che costituisce la stringa

- **Ad esempio, l'espressione "automobile".length() vale 10**

La stringa vuota

- **Una stringa vuota è una sequenza vuota di caratteri**
  - **la stringa vuota è un oggetto String di lunghezza zero**
  - **la stringa vuota è denotata dal letterale ""**
  - **l'espressione "".length() vale 0**

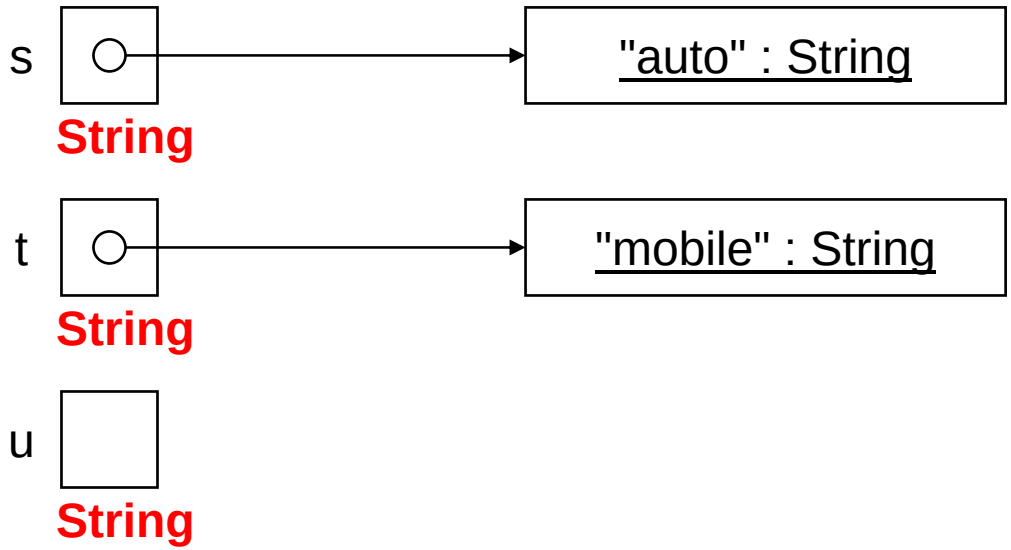
Il metodo String concat(String x)

- **La concatenazione è l'operazione che a partire da due stringhe S e T calcola una nuova stringa il cui valore è dato dalla sequenza di caratteri di S seguita dalla sequenza di caratteri di T**
  - il metodo **String concat(String x)** della classe **String** implementa l'operazione di concatenazione di stringhe
  - crea e restituisce un nuovo oggetto **String** composto dai caratteri della stringa su cui il metodo viene invocato seguiti dai caratteri della stringa argomento **x**

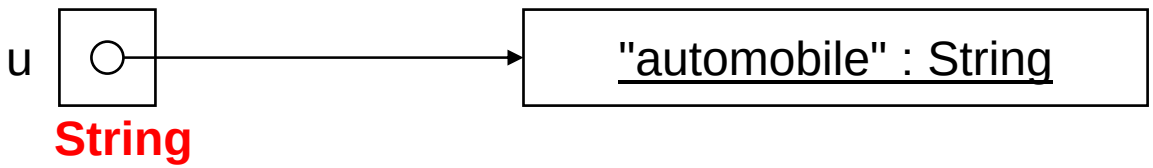
- **Ad esempio . . .**

Concatenazione di stringhe

```
String s, t, u;
s = "auto";
t = "mobile";
```




```
u = s.concat(t); // u vale (referenzia) "automobile"
```



Posizione dei caratteri in una stringa

- **Le posizioni dei caratteri di una stringa sono numerate da sinistra verso destra**
  - il carattere più a sinistra (il primo) ha posizione 0
  - in una stringa composta da N caratteri, il carattere più a destra (l'ultimo) ha posizione N – 1

    a u t o m o b i l e

    0 1 2 3 4 5 6 7 8 9

  - in generale, l'i-esimo carattere di una stringa occupa la posizione i–1

Il metodo String substring(int inizio, int fine)

- **Il metodo String substring(int inizio, int fine) della classe String**
  - (crea e) restituisce un nuovo oggetto String composto dai caratteri della stringa su cui il metodo è stato invocato che occupano le posizioni tra inizio (inclusa) e fine (esclusa)
    - detto in altro modo, la stringa restituita comprende i caratteri tra le posizioni inizio e fine–1 (estremi inclusi)
    - ad esempio

```
String s, t;
s = "automobile";
t = s.substring(2,6); // t vale "tomo"
```

Il metodo String substring(int inizio)

- **Il metodo String substring(int inizio) della classe String**
  - **(crea e) restituisce un nuovo oggetto String che consiste dei caratteri della stringa su cui il metodo viene invocato compresi tra quello di posizione inizio e l'ultimo carattere della stringa (incluso)**
  - **ad esempio**

```
String s, t;
s = "automobile";
t = s.substring(4); // t vale “mobile“
```

Pre-condizioni delle operazioni substring

- I metodi substring sono soggetti alle seguenti pre-condizioni
  - inizio deve essere maggiore o uguale a zero e minore o uguale alla lunghezza della stringa
  - fine deve essere minore o uguale alla lunghezza della stringa
  - inizio deve essere minore o uguale a fine
- In caso di violazione delle pre-condizioni viene generato un errore al tempo di esecuzione di tipo
  - IndexOutOfBoundsException
    - ovvero di tipo indice fuori dai limiti

Esempio: iniziali di un nome

```
public class JFK {
    public static void main(String[] args) {
      String  first = "John";
      String  middle = "Fitzgerald";
      String  last = "Kennedy";
      String  initials;
      String  firstInit, middleInit, lastInit;
      firstInit = first.substring(0,1);
      middleInit = middle.substring(0,1);
      lastInit = last.substring(0,1);
      initials = firstInit.concat(middleInit);
      initials = initials.concat(lastInit);
      System.out.println(initials);
    }
  }
```

Altri metodi della classe String

- La classe String ha molti metodi, consideriamone ora alcuni altri, altri ancora ne vedremo più avanti
  - String toUpperCase()
    - per convertire tutti i caratteri di una stringa in MAIUSCOLE
    - ad esempio `"Java".toUpperCase()` costruisce la stringa di valore JAVA
  - String toLowerCase()
    - per convertire tutti i caratteri di una stringa in minuscole
    - ad esempio `"Java".toLowerCase()` costruisce la stringa di valore java

Esempio: MaiuscoleMinuscole

```
public class UpperLowerCase {
    public static void main(String[] args) {
      String s, upper, lower;
      s = "Hello";
      upper = s.toUpperCase();
      lower = s.toLowerCase();
      System.out.println(s);
      System.out.print("upper = ");
      System.out.println(upper);
      System.out.print("lower = ");
      System.out.println(lower);
    }
  }
```

Questo programma costruisce 3 stringhe distinte (che non vengono più modificate): la stringa "Hello" iniziale, la stringa "HELLO" denotata attraverso upper, e la stringa "hello" denotata da lower

Cosa abbiamo visto finora

- Una introduzione a Java
- Cosa è una applicazione Java
- Cosa è il metodo main
- Alcuni primi esempi di applicazioni Java
  - Scrittore sullo schermo
  - Radice quadrata
- Alcuni primi esempi di classi e oggetti
  - Gli oggetti String con alcuni metodi e le loro possibili applicazioni

Riferimenti al libro di testo

- Per lo studio di questi argomenti si fa riferimento al libro di testo, e in particolare ai capitoli
  - 3 su Oggetti e Java
  - 7 su Leggibilità
  - 9 su Variabili e assegnazione
  - 12 sulle Stringhe
- In questa lezione abbiamo omesso gli argomenti trattati nei seguenti paragrafi
  - 3.2.3 – Perimetro di un triangolo
  - 3.2.4 – Lettura e somma di due numeri interi
  - 3.2.6 – Una classe per istanziare oggetti
  - Dal 12.5 al 12.10
    - Li vedremo più avanti
  - 3.2.5 – Robot in un labirinto
    - Che invece non farà parte del programma d'esame