FIGURE 2.5 Some Methods in the Class String

Method	Return Type	Example for String s = "Java";	Description
charAt (*index*)	char	c = s.charAt(2); // c='v'	Returns the character at *index* in the string. Index numbers begin at 0.
compareTo (*a_string*)	int	i = s.compareTo("C++"); // i is positive	Compares this string with *a_string* to see which comes first in lexicographic (alphabetic, with upper- before lowercase) ordering. Returns a negative integer if this string is first, zero if the two strings are equal, and a positive integer if *a_string* is first.
concat (*a_string*)	String	s2 = s.concat("rocks"); // s2 = "Javarocks"	Returns a new string with this string concatenated with *a_string*. You can use the + operator instead.
equals (*a_string*)	boolean	b = s.equals("Java"); // b = true	Returns true if this string and *a_string* are equal. Otherwise returns false.
equals IgnoreCase (*a_string*)	boolean	b = s.equals("java"); // b = true	Returns true if this string and *a_string* are equal, considering upper- and lowercase versions of a letter to be the same. Otherwise returns false.
indexOf (*a_string*)	int	i = s.indexOf("va"); // i = 2	Returns the index of the first occurrence of the substring *a_string* within this string or –1 if *a_string* is not found. Index numbers begin at 0.
lastIndexOf (*a_string*)	int	i = s.lastIndexOf("a"); // i = 3	Returns the index of the last occurrence of the substring *a_string* within this string or –1 if *a_string* is not found. Index numbers begin at 0.
length()	int	i = s.length(); // i = 4	Returns the length of this string.
toLower Case()	String	s2 = s.toLowerCase(); // s2 = "java"	Returns a new string having the same characters as this string, but with any uppercase letters converted to lowercase. This string is unchanged.
toUpper Case()	String	s2 = s.toUpperCase(); // s2 = "JAVA"	Returns a new string having the same characters as this string, but with any lowercase letters converted to uppercase. This string is unchanged.

replace (*oldchar, newchar*)	String	s2 = s.replace('a','o');
		// s2 = "Jovo";
			Returns a new string having the same characters as this string, but with each occurrence of *oldchar* replaced by *newchar*.

substring (*start*)	String	s2 = s.substring(2);
		// s2 = "va";
			Returns a new string having the same characters as the substring that begins at index *start* through to the end of the string. Index numbers begin at 0.

substring (*start,end*)	String	s2 = s.substring(1,3);
		// s2 = "av";
			Returns a new string having the same characters as the substring that begins at index *start* through to but not including the character at index *end*. Index numbers begin at 0.

trim( )	String	s = " Java ";
		s2 = s.trim();
		// s2 = "Java"
			Returns a new string having the same characters as this string, but with leading and trailing whitespace removed.