Java Programming

Strings

String Variables

- A **String** in Java is actually a class in the Java Class Library. It can be used to hold just about any line of text.

String Declaration

- In code **String** reference variables are defined using the class name **String** followed by the variable name.
- Names for **String** variables follow the standard Java rules for variable names.

- For example:

```
String s;
String str;
String myName;
```

Creating Instances of Strings & Storing Values

- Instances of Strings can be created by using the **new** operator or just by setting a **String** variable to some text in double quotes.
- This actually instantiates an instance of the String class and then stores the string in it.

- For example:

```
s = new String();    // Instantiate a new String
s = "My string";     // Store "My string" in the String instance s

// Both lines below first instantiate a new String object and then
// store the shown text in it.  In the first an instance of String
// is explicitly created using the new operator. In the second the
// instance of String is automatically instantiated just by using
// the equals to store a string in it.

str = new String("This is a String.");
myName = "Able Programmer";
```

String Functions

- Because the **String** class is part of the Java Library, there are quite a few built-in functions you can use with any instance of **String** that you create in a program.

String Function Examples

- In the examples below the **String** myName has "Able Programmer" stored in it.

- **char charAt(int index)** -- Returns the character at the given index. The first character in a string is at index 0.
  - Example: char ch = myName.charAt(5) stores 'P' in the char variable ch.

- **int compareTo(String str)** -- Compare str to the String If str comes before the String alphabetically then a negative number is returned. If str is identical to the String then zero is returned. If str comes after the String alphabetically then a positive number is returned.
  - Example: int val = myName.compareTo("Aardvark") sets val to a negative number ("Aardvark" alphabetically comes before "Able Programmer", int val = myName.compareTo("Able Programmer") sets val to zero, and int val = myName.compareTo("Better than average") sets val to a positive number.

- **int compareToIgnoreCase(String str)** -- Works exactly like **compareTo** except the case of letters is ignored.
  - Example: int val = myName.compareTo("ABLE PROGRAMMER") sets val to zero even though one is all upper case letters and the other is upper and lower case letters.

- **String concat(String str)** -- Concatenates (appends) str onto the end of the given string.
  - Example: myName.concat(", Ph.D") makes the string stored in myName now read "Able Programmer, Ph.D." Looks like we will now have to refer to Able as Dr. Programmer.

- **int indexOf(char c)** -- Returns the index of the first occurance of a given character in the String.
  - Example: int loc = myName.indexOf('e') sets loc to 3 (the index of the 'e' in "Able").

- **int indexOf(char c, int fromIdx)** -- Returns the index of the first occurance of a given character in the String starting from index fromIdx.
  - Example: int loc = myName.indexOf('e', 4) sets loc to 13 (the index of the 'e' at the end of "Programmer").

- **int indexOf(String s)** -- Returns the index of the first occurance of a given String in the String.
  - Example: int loc = myName.indexOf("Pro") sets loc to 5.

More String Function Examples

- In the examples below the **String** myName has "Able Programmer" stored in it.
- **int indexOf(String s, int fromIdx)** -- Returns the index of the first occurance of a given String in the String starting from index fromIdx.
  - Example: int loc = myName.indexOf("Pro", 4) sets loc to 5 (the same as indexOf, but the search started at the space after "Able").
- **int lastIndexOf(char c)** -- Returns the index of the last occurance of a given character in the String.
  - Example: int loc = myName.lastIndexOf('e') sets loc to 13 (the index of the 'e' at the end of "Programmer").
- **int lastIndexOf(char c, int fromIdx)** -- Returns the index of the last occurance of a given character in the String starting from index fromIdx.
  - Example: int loc = myName.lastIndexOf('e', 12) sets loc to 13 (the index of the 'e' in "Able"). The search starts at the 'm' before the 'e' in "Programmer" and searches backward.
- **int lastIndexOf(String s)** -- Returns the index of the last occurance of a given String in the String.
  - Example: int loc = myName.lastIndexOf("Pro") sets loc to 5, the same as the results of calling indexOf, but the search was done from the end of the string.
- **int lastIndexOf(String s, int fromIdx)** -- Returns the index of the last occurance of a given String in the String starting from index fromIdx.
  - Example: int loc = myName.lastIndexOf("Pro", 12) sets loc to 5 (the same as lastIndexOf, but the search started at the 'm' just before the 'e' in "Programmer" searching backward).
- **int length()** -- Returns the number of characters in the String
  - Example: int len = myName.length() sets len to 14.

Even More String Function Examples

- In the examples below the **String** myName has "Able Programmer" stored in it.
- **String substring(int beginIdx)** -- Returns a string that is a substring of the given String starting at the given index.
  - Example: String ss1 = myName.substring(5) sets ss1 to "Programmer".
- **String substring(int beginIdx, int endIdx)** -- Returns a string that is a substring of the given String starting at beginIdx up to, but not including the endIdx.
  - Example: String ss2 = myName.substring(5, 8) sets ss1 to "Pro".
- **String toLower()** -- Converts all of the letters in a string to lower case.
  - Example: myName.toLower() sets myName to "able programmer".
- **String toUpper()** -- Converts all of the letters in a string to upper case.
  - Example: myName.toUpper() sets myName to "ABLE PROGRAMMER".
- **String valueOf(double val)** -- Creates a string from the given double value. This is a one of several special functions that can be called directly using the **String** class name without having to have an actual instance of a string.
  - Example: String str = String.valueOf(3.14159) sets str to the string "3.14159". You can also do the same thing with int, float, and long values.