

The String Type

- A string is a sequence of characters
- The String type is used to declare variables that store strings
- The String type is not a *primitive* type: it is known as a *class* or *reference* type
- A String constant is one or more characters in double quotes, e.g., "string constant"
- Examples:

```
char charVariable = 'a'; //single quotes
String stringVariable = "a"; //double quotes
String sentence = "Hello, world";
```

String Variables

- Declare a String variable:

```
String greeting;
```
- Assign a value to the variable:

```
greeting = "Hello!";
```
- Use the variable as a String argument in a *method* call:

```
System.out.println(greeting);
```

causes the string Hello! to be displayed on the screen

Indexing Characters Within a String

- The index of a character within a string is an integer starting at 0 for the first character and gives the position of the character
- For example:

```
String str = "This is a string";
```

T	h	i	s		i	s		a		s	t	r	i	n	g
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15

Methods

- A *method* is an operation with a name and a list of arguments (possibly empty)
- A method can simply perform an action or it can return a value
- A call to a method that does not return a value is a *statement*, e.g.,
`System.out.println("Hi there!");`
- A call to a method that returns a value is an *expression*, e.g.,
`int i = keyboard.nextInt();`

Some String Methods

- The String type has many methods that allow us to manipulate String values/variables
- String methods are called/invoked with the following syntax:

```
stringVar.methodName( arguments )
```

Some String Methods

- **int** length()
returns the number of characters in the given string, e.g.,

```
String str = "This is a string";  
System.out.println(str.length());
```

What's the output?

String Methods cont.

- **char** `charAt(int pos)`
returns the character at position *pos* in the given string, e.g.,

```
String str = "This is a string";  
System.out.println(str.charAt(0));  
System.out.println(str.charAt(1));  
System.out.println(str.charAt(15));
```

What's the output?

Some String Methods cont.

- **String** `substring(int start, int end)`
returns the string starting at position *start* and ending at position (*end-1*) in the given string, e.g.,

```
String str = "This is a string";  
System.out.println(str.substring(0,4));  
System.out.println(str.substring(5,7));  
System.out.println(str.substring(0,16));
```

What's the output?

Some String Methods cont.

- `int indexOf(String aString)`
returns the position of the first occurrence of string *aString* in the given string (or -1 if not found), e.g.,

```
String str = "This is a string";
System.out.println(str.indexOf("This"));
System.out.println(str.indexOf("is"));
System.out.println(str.indexOf("yoh"));
```

What's the output?

Concatenating (Appending) Strings

- As we have seen before the `+` operator can be used to concatenate string values, e.g.,

```
String name = "Cindy";
String greeting = "Hi, " + name + "!";
System.out.println(greeting);
```

What is the output?

Single Character Input

Given the import:

```
import java.util.Scanner;
```

and the declaration:

```
Scanner in = new Scanner(System.in);
```

- Declare and input a single character:

```
String s = in.nextLine();
```

```
char c = s.charAt(0);
```

- Note: actually, input a whole line.

Escape Characters

- How do you print the following string?

```
The word "hard"
```

- Would this do it?

```
System.out.println("The word "hard");
```

- No, it would give a compiler error - it sees the string
The word between the first set of double quotes
and is confused by what comes after

- Use the backslash character, "\", to escape the special meaning of the internal double quotes:

```
System.out.println("The word \"hard\");
```

String Program

```
public class StringTest
{
    public static void main(String[] args)
    {
        String greeting = "Hello!";
        int len = greeting.length();
        System.out.println("Length is " + len);
        char ch = greeting.charAt(3);
        System.out.println("Character at position 3 is " + ch);
        String sub = greeting.substring(1, 3);
        System.out.println("Substring[1..3] is " + sub);
        int index1 = greeting.indexOf("lo");
        System.out.println("Index of \"lo\" is " + index1);
        int index2 = greeting.indexOf("low");
        System.out.println("Index of \"low\" is " + index2);
    }
}
```

What Is The Output Of StringTest?

- Trace through the statements of StringTest and determine the output produced by the program.

Your Turn, Again!

- Write a Java program called *BreakName*, which asks the user for his/her name in the form *First M. Last*, and outputs First, M., and Last on three different lines.
- In other words, after the name is read from input, the program needs to break it up in the three pieces (First, M., and Last) and output those one line at a time.

BreakName

Documentation and Style

- Use meaningful names for variables, programs, etc.
- Use indentation and line spacing as shown in the examples in the text
- Always include a “prologue” (a brief explanation of the program at the beginning of the file)
- Use all lower case for variables, except capitalize internal words (`eggsPerBasket`)

Comments

- *Comment*—text in a program that the compiler ignores
- Does not change what the program does, only explains the program
- Write meaningful and useful comments
- Comment the *non-obvious*
- Assume a *reasonably* knowledgeable reader
- `//` for single-line comments
- `/* ... */` for multi-line comments