# Class, not Primitive

## Manipulating characters and groups of characters provides some challenges

## A String is a class

### Each created String is a class object

### The String variable name is not a simple data type

### **Reference**

#### A variable that holds a memory address

## Compare two Strings using the == operator

### Not comparing values

### Comparing computer memory locations

## Compare contents of memory locations more frequently than memory locations themselves

## Classes to use when working with character data

### **Character**

#### Instances hold a single character value

#### Defines methods that can manipulate or inspect single-character data

### **String**

#### A class for working with fixed-string data

##### Unchanging data composed of multiple characters

# When Many Changes

#### Classes for storing and manipulating changeable data composed of multiple characters

## Character class

### Contains standard methods for testing the values of characters

### Methods that begin with "is"

#### Such as isUpperCase()

#### Return a Boolean value that can be used in comparison statements

### Methods that begin with "to"

#### Such as toUpperCase()

#### Return a character that has been converted to the stated format

## Literal string

### A sequence of characters enclosed within double quotation marks

### An unnamed object, or **anonymous object**, of the String class

## **String variable**

### A named object of the String class

## Class String

### Defined in java.lang.String

### Automatically imported into every program

## Declare a String variable

### The String itself is distinct from the variable used to refer to it

### You can create a String object without:

#### Using the keyword new

#### Explicitly calling the class constructor

## String is a class

### Each created String is a class object

## String variable name

### A reference variable

### Refers to a location in memory

#### Rather than to a particular value

## Assign a new value to a String

### The address held by the String is altered

## **Immutable**

### Objects that cannot be changed, such as a String

## Making simple comparisons between Strings often produces misleading results

## Comparing Strings using the == operator

### Compares memory addresses, not values

## **equals() method**

### Evaluates the contents of two String objects to determine if they are equivalent

### Returns true if objects have **identical** contents

### public boolean equals(String s)

## **equalsIgnoreCase() method**

### **Ignores case** when determining if two Strings are equivalent

### Useful when users type responses to prompts in programs

## **compareTo() method**

### Compares two Strings and returns:

#### **Zero**: If two Strings refer to the same value

#### **Negative number**: If the **calling object** is "less than" the argument

#### **Positive number**: If the **calling object** is "more than" the argument

### if (**aWord**.compareTo(anotherWord) < 0)

## Empty Strings **" "**

## Reference a memory address with no characters

### Can be used in String methods

## **null Strings**

### Use the null Java keyword

### Strings are set to null by default

### Cannot be used in String methods

#### Cause an error

### No address

## **toUpperCase()** and **toLowerCase()** methods

### Convert any String to its uppercase or lowercase equivalent

## **length() method**

### Returns the length of a String

### Start at 0

## **indexOf() method**

### Determines whether a specific character occurs within a String

### Returns the position of the character

### The first position of a String is zero

### The return value is –1 if the character does not exist in the String

## **charAt() method**

### Requires an integer argument

### Indicates the position of the character that the method returns

## **endsWith() method** and **startsWith() method**

### Each takes a String argument

### Return true or false if a String object does or does not end or start with the specified argument, respectively

## **replace() method**

### Replaces all occurrences of some character within a String

## **toString() method**

### Not part of the String class

### We use it with print() and println()

### Converts any object to a String

## **Concatenation**

### Join a simple variable to a String

### String aString = "My age is " + myAge;

### Use the + operator

## **substring() method**

### Extracts part of a String

### Takes two integer arguments

#### Start position

#### End position

### The length of the extracted substring is the difference between the second integer and the first integer

## regionMatches() method

### Two variants that can be used to test if two String regions are equal

## A substring of the specified String object is compared to a substring of the other

### If the substrings contain the same character sequence, then the expression is true

### Otherwise, the expression is false

## A second version uses an additional boolean argument

### Determines whether case is ignored when comparing characters

## **Integer class**

### Part of java.lang

### Automatically imported into programs

### Converts a String to an integer

### **parseInt() method**

#### Takes a String argument

#### Returns its integer value

## **Wrapper**

### A class or an object "wrapped around" a simpler element

## Integer class valueOf() method

### Converts a String to an Integer class object

## Integer class intValue() method

### Extracts the simple integer from its wrapper class

## **Double class**

### A wrapper class

### Imported into programs automatically

### **parseDouble() method**

#### Takes a String argument and returns its double value

## The value of a String is fixed

### After a String is created, it is immutable

### If you expect to change string quite a bit, use it

## **StringBuilder** and **StringBuffer** classes

### An alternative to the String class

### Used when a String will be modified

### Can use anywhere you would use a String

### Part of the java.lang package

### Automatically imported into every program

## StringBuilde

### More efficient

## StringBuffer

### Thread safe

### Use in multithreaded programs

# StringBuilder Object

## Create a StringBuilder object

### StringBuilder eventString = new StringBuilder ("Hello there");

### Must use:

#### The keyword new

#### The constructor name

#### An initializing value between the constructor's parentheses

# What is the Buffer?

## **Buffer**

### A memory block

### Might or might not contain a String

### The String might not occupy the entire buffer

#### The length of a String can be different from the length of the buffer

### **Capacity**

#### The actual length of the buffer

## **setLength() method**

### Changes the length of a String in a StringBuilder object

## length property

### An attribute of the StringBuilder class

### Identifies the number of characters in the String contained in the StringBuilder

## **capacity() method**

### Finds the capacity of a StringBuilder object

## Using StringBuilder objects

### Provides improved computer performance over String objects

### Can insert or append new contents into StringBuilder

## **append() method**

### Adds characters to the end of a StringBuilder object

## **insert() method**

### Adds characters at a specific location within a StringBuilder object

## **setCharAt() method**

### Changes a character at a specified position within a StringBuilder object

# And more…

## **charAt() method**

### Accepts an argument that is the offset of the character position from the beginning of a String

### Returns the character at that position

# Summary\_0

## String variables

### References

## Character class

### Instances can hold a single character value

## Each String class object

### Is immutable

### equals() method

## toString() method

### Converts any object to a String

# Summary\_1

## Integer.parseInt() method

### Takes a String argument and returns an integer value

## Double.parseDouble() method

### Takes a String argument and returns a double value

## StringBuilder or StringBuffer class

### Improves performance when a string's contents must change