Character and Strings

Colorado State University

Department of Computer Science

Slides Originally Created by Albert Lionelle (Albert.Lionelle@colostate.edu) ,
updated by Marcia Moraes (marcia.moraes@colostate.edu)

Announcements

- Reminder – readings are due **before** lecture
  - You don’t have to do all of it - challenge problems can be challenging…
  - You can return to them.
  - We start off each lecture with a recall activity!
- Help Sessions
  - Go to them! They make a difference
  - Lab teams should be setup
    - If you are not on a private channel with your other lab mates, let us know
- Thursday lab – meant to be done by Monday

Recall Activity

Grab a paper, write your name and your answers to the following questions. Turn this as your attendance for today's lecture.

What is an object?

What is the difference between a char and a String?

Given the following Code, what is printed?

```
public static char characterAddition(char a, int b) {
    return (char) (a + b);
}

public static void main(String[] args) {
    System.out.println(characterAddition('A', 2));
}
```

C

Characters (char)

- primitive - stores **values** only
- Example declarations:

```
char myChar = 'x'; // notice single quotes
char newLine = '\n'; //the invisible newline character
char myChar2 = 57; // bad idea! It sets the value to the character ‘9’
char what = (char)('A' + 2);  // what is now ‘C’ (used in things like the Caesar cipher)
```

- A char is an int behind the scenes - and java uses the ASCII Table to figure out the character value

Casting

- Type casting is when you assign a value of one primitive data type to another type.
  - This allows you to change an int back to a char without an error!

```
public static char characterAddition(char a, int b) {
    return (char) (a + b);
}

public static void main(String[] args) {
    System.out.println(characterAddition('A', 2));
}
```

- a+b
  - Converts **a** to an int (65)
  - adds **b** which is 2, so 67 total
- **(char)** takes 67, and makes it ‘C’, a char
- **‘C’** gets returned

Quick Practice

```
public class CastingProgram {

    public static void main(String[] args) {

        int value1 = 12;
        double value2 = 25.75;
        double resultDouble = value1 + value2;
        System.out.print(value1 + " + " + value2 + " = ");
        System.out.println(resultDouble);
        int resultInt = value1 +  value2;
        System.out.print(value1 + " + " + value2 + " = ");
        System.out.println(resultInt);
    }
}
```

What happen when we try to run the program?

CastingProgram.java:10: error: incompatible types: possible lossy conversion from double to int
 int resultInt = value1 + value2; ^ 1 error

How to solve that error?

Quick Practice

```
public class CastingProgram {

    public static void main(String[] args) {

        int value1 = 12;
        double value2 = 25.75;
        double resultDouble = value1 + value2;
        System.out.print(value1 + " + " + value2 + " = ");
        System.out.println(resultDouble);
        int resultInt = value1 +  (int) value2;
        System.out.print(value1 + " + " + (int) value2 + " = ");
        System.out.println(resultInt);
    }
}
```

Manually casting the double variable to an int

Hang’em

- Time play Hang’em
- In your groups, play a game of hangman
  - Try to use Java syntax words
  - Try multiple word sentences
  - Include the *space* as one of the characters in a multiword sentence

- Ok - next part, write numbers below each dash
  - start at 0
  - make sure to number (index!) the spaces

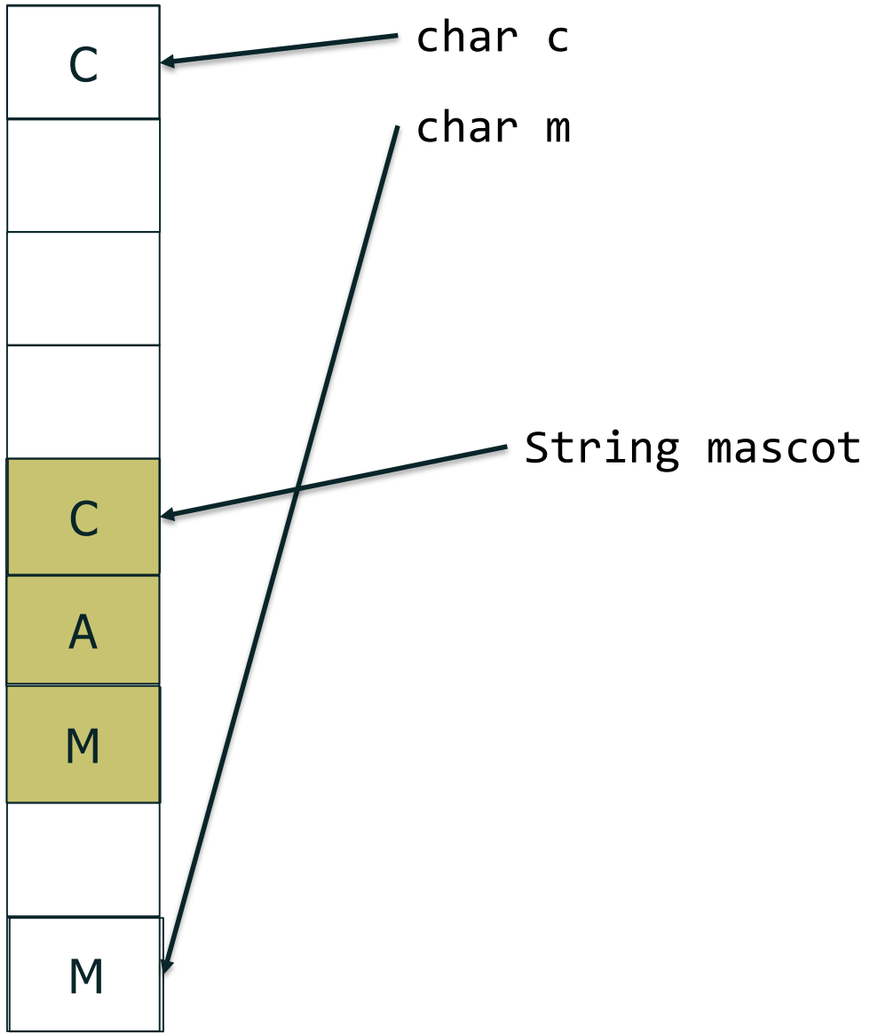
```
H e l l o _ J a v a
0 1 2 3 4 5 6 7 8 9
```

- You just created a java String

Let’s talk about String

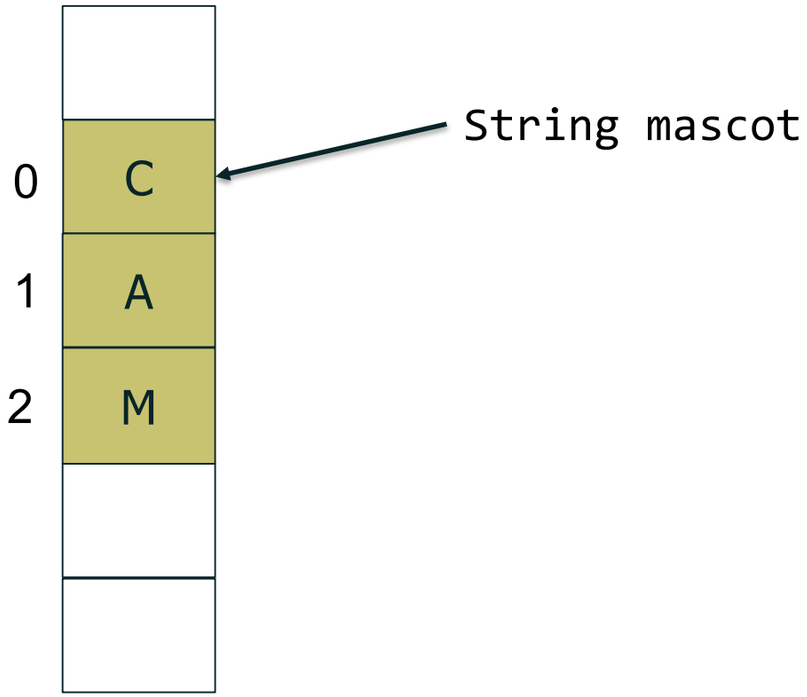
- A String is a collection of ordered characters
  - It has data
  - It has functionality (methods)
  - It is also **immutable** (can’t be directly modified)
    - Every method that builds a String, returns a copy
    - Will cover this more in the future
- But what does a String really look like?

First, Let's Think About Memory



- Storing variables in memory
  - Wherever the computer decides
  - Value stored in the location
  - Variable name points towards location
- Example
  - `char c = 'C';`
  - `char m = 'M';`
- What about Strings?
  - `String mascot = "CAM";`

String Methods



- Ordered Collection of characters
  - Indexed – starting at 0
  - 'C' is at index 0
  - 'A' is at index 1
- **`char character = mascot.charAt(2);`**
  - returns 'M' (character)
- How many characters?
  - **`int len = mascot.length();`** `// 3`
- No matter how large the string
  - Starts at 0
  - Has a length
  - Always know first and last index!

Making Strings Look Good

Reading Check-in

Given the following Code, what is printed?

```
public static String characterAddition(String a, int b) {
    return a + b;
}

public static void main(String[] args) {
    System.out.println(characterAddition("A", 2));
}
```

A2

String Concatenation

- Adds two strings together
  - Better worded: String objects added to other types (String or primitive or other objects)
- Step 1 converts everything to a String. So 1 number becomes "1"
- Step 2 puts it all together, exactly as written.

Example:

```
int aNumber = 42;
String question = "The Answer To Life, The Universe And Everything";
System.out.println(question + aNumber);
```

Prints "The Answer To Life, The Universe And Everything42" .. whoops

```
System.out.println(question + "  is " + aNumber);
```

Prints "The Answer To Life, The Universe And Everything is 42"

String.format/printf

- We know we can write Strings via Concatenation

```
String towelColor = "pink";
String dontPanic = "Don't forget to bring your " + towelColor + " towel";
String txt = "Don't panic";
String heading1  = "<h1>" + txt + "</h1>";
```

- However, that can be awkward - introducing **String.format**

```
String towelColor = "pink";
String dontPanic = String.format("Don't forget to bring your %s towel", towelColor);
String txt = "Don't panic“;
String heading1 = String.format("<h1>%s</h1>", txt);
```

- **printf** is `System.out.printf("<h1>%s</h1>", txt);` // prints <h1>Don’t panic</h1> to the screen

Example

```
public static String businessCardNoFormat(String name) {
   return  "+==================================+\n" +
           "|  Round Table Enterprises         |\n" +
           "|                                  |\n" +
           "|            " + name +" |\n" +
           "+==================================+";
}
public static String businessCardUsingFormat(String name) {
   return  "+==================================+\n" +
           "|  Round Table Enterprises         |\n" +
           "|                                  |\n" +
            String.format("|%33s |\n", name) +
           "+==================================+";
}
```

Without String.format

```
+==================================+
|  Round Table Enterprises         |
|                                  |
|            Lancelot of the Lake  |
+==================================+
```

With String.format

```
+==================================+
|  Round Table Enterprises         |
|                                  |
|                Bors the Younger  |
+==================================+
```

Numbers in String format

```
public static String percentNoFormat(double val) {
    return val + "%";
}
```

33.333333333333%

```
public static String percentFormat(double val) {
    return String.format("%.2f%%", val);
}
```

33.33%

String.format References

- The % character is followed by special characters
  - %s for string, %f for floating point, etc
  - Capitalize the %S or %C to force uppercase!
- You can specify the number of decimal places by including the numbers
    - %.2f // example
    - String.format("%.2f", 100)
    - would produce 100.00
- You can also do cool things like pad variables, etc
  - Search for **printf()** online
    - Very common in over 20+ languages
  - Admittedly, I look up a lot of the formats as I need them
    - %s, %S, %d, %0.2f, %n
    - Those are good to know off the top of your head
  - %n == new line character
  - %% == % sign (since you can't just type %)

Additional Reading: Detailed Format Reference

Quick Practice

```
public class StringFormatProgram {
    public static void main(String[] args) {
        int value1 = 12;
        double value2 = 25.75986;
        String color = "red";
        int len = color.length();
        System.out.println("Length of color: " + len);
        System.out.println("First letter of color: " + color.charAt(0));
        System.out.printf("%d\n", value1);
        System.out.printf("%.2f\n", value2);
        System.out.printf("%20.2f\n", value2);
        System.out.printf("%s\n", color);
        System.out.printf("%S\n", color);
    }
}
```

What is the exactly output of this program?

Change the System.out.println to System.out.printf

Print the last character that is stored in the color variable