INFO 5100

Application Engineering Design

String class

Daniel Peters

d.peters@neu.edu

- Lecture:
  1. Java String class
  2. Java StringBuilder class

Java String

- Character String

  "This is a LITERAL character string."

- A String is a Reference Type

  java.lang.String class

- A String is immutable
- **NOT** an array of characters terminated by a null character (C Language).
  - A Java String object is **NOT** a C language string.

Java String

- Special String class treatment:
  - Enclosing characters in double quotes **automatically** creates a String object:

String **name** = "**Dan**";

  - Identifier "**name**" contains a ***reference*** to a String object containing the immutable value of "**Dan**".

Java String

- For String objects, Use of the ‘new’ keyword is optional (and discouraged)
  - Reference: Java **String pool** and **String interning**.
  - Both memory (and its allocation time) are conserved by saving immutable Strings in a pool. When a new String is created, **if it is a repeated String**, a reference to an already preserved immutable String in the pool is established in lieu of a new String created.

Java String

- Use of the ‘new’ keyword is optional (and discouraged) for creating String objects.
- DO

  String s = “abc”; // allows interning
- DO NOT

  String s = new String(“abs”); // forces new string

- String objects

String Operations

- String.**toUpperCase**()
  - Converts String to ALL CAPS
- String.**toLowerCase**()
  - Convert String to ALL LOWERCASE
- Compare: s1.**compareTo**(s2);
  - Returns 0 indicating lexagraphically equal strings

String Split

- String split

  String s = **"Dan,17,4.0";**

  String [] tokens = s.**split**(",");

  System.***out.println("Student:"***

  + " NAME: " + tokens[0]

  + ", AGE: " + tokens[1]

  + ", GPA: “ + tokens[2]);

- OUTPUT:

Student: NAME: Dan, AGE: 16, GPA: 4.0

String to Integer Conversion

- **Integer**.parseInt()
  - Convert String to int value

```
String s = “17”;          // String representation of int 17
int age = 0;
try {
        age = Integer.parseInt(s); // convert String to int
} catch (NumberFormatException e) {
        System.out.println(s + " is not a number!");
        e.printStackTrace();
}
System.out.println(s + “ is Age: “ + age);
```

String to Integer Conversion

- CONSOLE OUTPUT:

17 is Age: 17

String to Double Conversion

- **Double**.parseDouble()
  - Convert String to double value

```
String s = “4.0”;          // String representation of 4.0
double gpa= 0;
try {
        gpa = Double.parseDouble(s); // String to double
} catch (NumberFormatException e) {
        System.out.println(s + " is not a  number!");
        e.printStackTrace();
}
System.out.println(s + “ is GPA: “ + gpa);
```

String to Double Conversion

- CONSOLE OUTPUT:

4.0 is GPA: 4.0

SubString

- SubString

```
String s = new String ("abcd");
int ix1 = 1;
int ix2 = 3;
sub = s.substring(ix1);          // bcd
sub = s.substring(ix1,ix2); // bc
int ix1 = 0;
sub = s.substring(ix1,ix2); // abc
```

String Concatenation

```
String sb = "Peter” +
+ ",” +
+ "Paul” +
+ ",” +
+ "Mary” +
+ ",” ;
System.out.println(sb);
```

StringBuilder

```
StringBuilder sb = new
StringBuilder("Peter");
sb.append(",");
sb.append("Paul");
sb.append(",");
sb.append("Mary");
sb.append(",");
System.out.println(sb.toString());
```

StringBuilder

- CONSOLE OUTPUT:

Peter, Paul, Mary,