

More on Strings

String methods and equality

Produced Mairead Meagher
by: Dr. Siobhán Drohan



Waterford Institute *of* Technology
INSTITIÚID TEICNEOLAÍOCHTA PHORT LÁIRGE

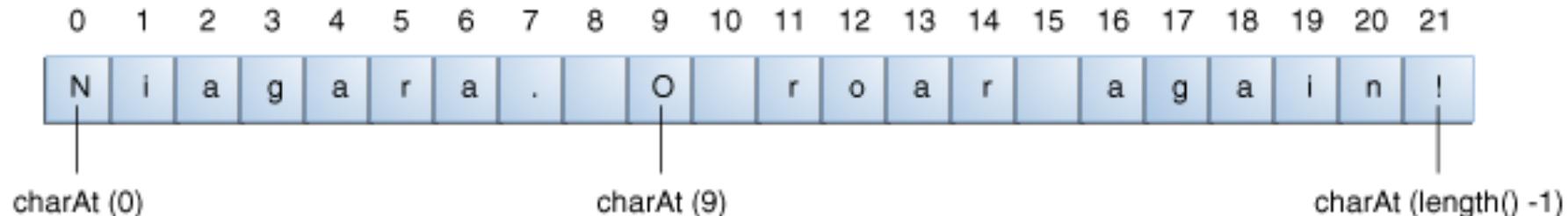
Department of Computing and Mathematics
<http://www.wit.ie/>

Topics list

- Strings: index of characters
- String methods:
 - `charAt(int index)`
 - `substring(int beginIndex, int endIndex)`
 - `compareTo(String anotherString)`
- Recap: Primitive vs object
- String Identity versus equality
- null
- Escape Sequences

Strings: index of characters

- A String holds a sequence of characters.
- The index of the first character in a String is 0.
- The index of the last character in a String is `length()-1`.



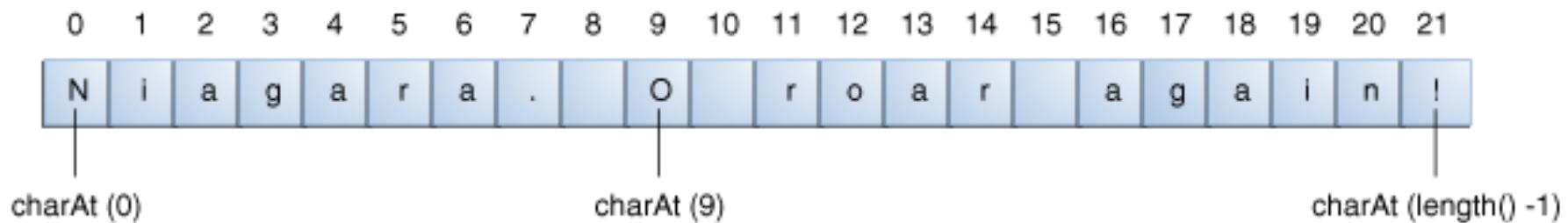
Topics list

- Strings: index of characters
- String methods:
 - `charAt(int index)`
 - `substring(int beginIndex, int endIndex)`
 - `compareTo(String anotherString)`
- Recap: Primitive vs object
- String Identity versus equality
- null
- Escape Sequences

String methods: charAt(int index)

- The following code gets the character at index 9 in a String:

```
String anotherPalindrome = "Niagara. O roar again!";
char aChar = anotherPalindrome.charAt(9);
```



Indices begin at 0, so the character at index 9 is 'O'

Processing example 7.1

The screenshot shows the Processing 2.2.1 IDE interface. The title bar says "sketch_151021a | Processing 2.2.1". The menu bar includes File, Edit, Sketch, Tools, and Help. Below the menu is a toolbar with icons for play, stop, save, and run, followed by a "Run" button. The sketch window contains the following Java code:

```
String alphabet = "abcdefghijklmnopqrstuvwxyz";
String errorMessage404 = "HTTP 404 Not Found Error";

println("The character at position 4 in "
    + alphabet
    + " is "
    + alphabet.charAt(3));

println("The character at position 10 in "
    + errorMessage404
    + " is "
    + errorMessage404.charAt(9));
```

The output window at the bottom displays the results of the println statements:

```
The character at position 4 in abcdefghijklmnopqrstuvwxyz is d
The character at position 10 in HTTP 404 Not Found Error is N
```

Finding the character located a specific position in a String.

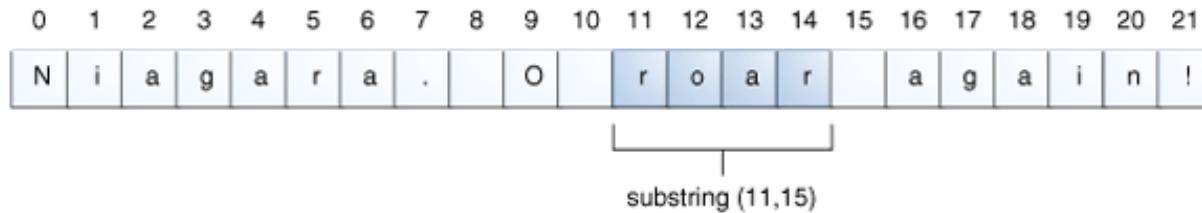
Topics list

- Strings: index of characters
- String methods:
 - `charAt(int index)`
 - `substring(int beginIndex, int endIndex)`
 - `compareTo(String anotherString)`
- Recap: Primitive vs object
- String Identity versus equality
- null
- Escape Sequences

String methods: substring(int beginIndex, int endIndex)

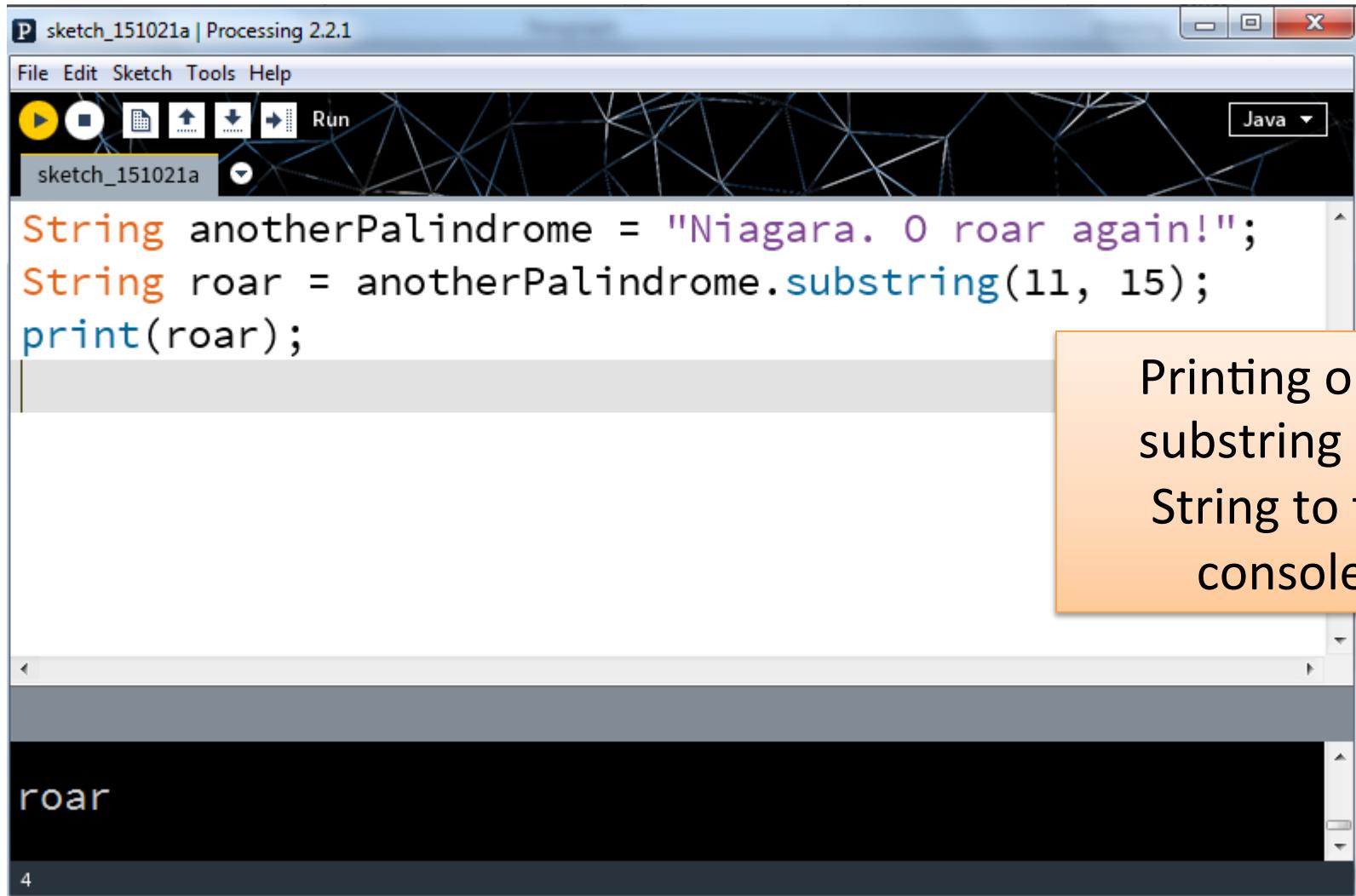
- This method returns a new String that is a substring of this String.
- The substring begins at the specified beginIndex and extends to the character at index endIndex – 1

```
String anotherPalindrome = "Niagara. O roar again!";
String roar = anotherPalindrome.substring(11, 15);
```



This code returns a substring ("roar") from anotherPalindrome. It extends from index 11 up to, but not including, index 15.

Processing example 7.2, version 1

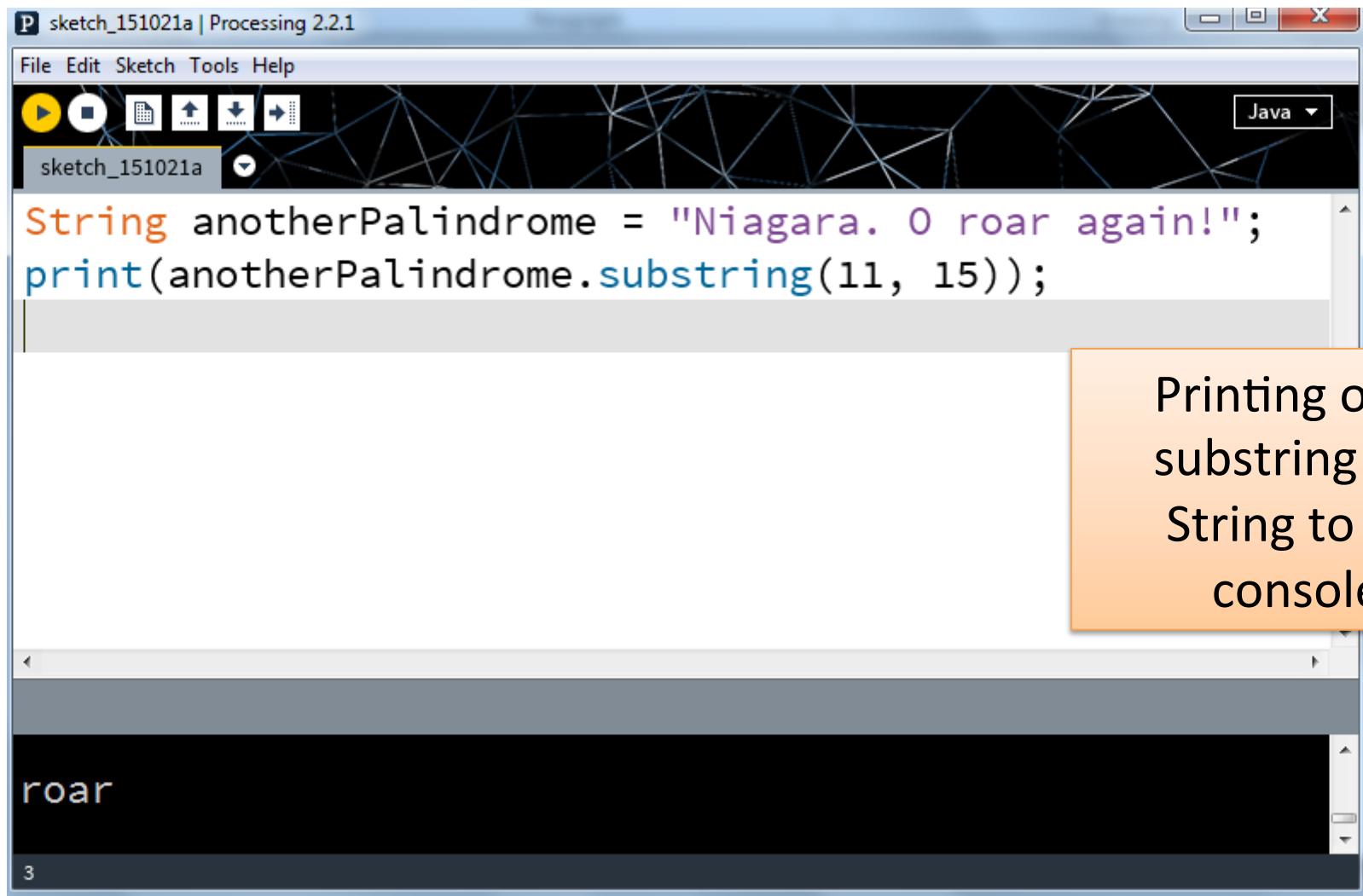


The screenshot shows the Processing 2.2.1 IDE interface. The title bar reads "sketch_151021a | Processing 2.2.1". The menu bar includes File, Edit, Sketch, Tools, and Help. The toolbar contains icons for play, stop, save, and run, with "Run" being highlighted. A dropdown menu shows "Java". The code editor window displays the following Java code:

```
String anotherPalindrome = "Niagara. O roar again!";
String roar = anotherPalindrome.substring(11, 15);
print(roar);
```

The output window at the bottom shows the result "roar". An orange callout box on the right side of the code editor contains the text: "Printing out a substring of a String to the console."

Processing example 7.2, version 2



The screenshot shows the Processing 2.2.1 IDE interface. The title bar reads "sketch_151021a | Processing 2.2.1". The menu bar includes File, Edit, Sketch, Tools, and Help. The toolbar contains icons for play, stop, save, and load. The sketch name "sketch_151021a" is displayed in the center toolbar. A dropdown menu shows "Java". The code editor contains the following Java code:

```
String anotherPalindrome = "Niagara. 0 roar again!";
print(anotherPalindrome.substring(11, 15));
```

The output window at the bottom shows the result "roar". An orange callout box on the right side of the code editor contains the text:

Printing out a
substring of a
String to the
console.

Topics list

- Strings: index of characters
- String methods:
 - `charAt(int index)`
 - `substring(int beginIndex, int endIndex)`
 - `compareTo(String anotherString)`
- Recap: Primitive vs object
- String Identity versus equality
- null
- Escape Sequences

String methods: compareTo

`int compareTo (String anotherString)`

- This method compares two strings lexicographically i.e. based on the Unicode value of the characters in the String.
- It returns an integer indicating whether this string is:
 - greater than (result is > 0)
 - equal to (result is $= 0$) or
 - less than (result is < 0) the argument, anotherString.

Processing example 7.3

```
String str1 = "Dog";
String str2 = "Cat";

if(str1.compareTo(str2) < 0){
    println(str1+" comes before "+str2 +" in the alphabet");
}
else if(str1.compareTo(str2) > 0){
    println(str2 + " comes before "+str1+" in the alphabet");
}
else{
    println("The strings are identical");
}
```

What will be printed to the console?
Which boolean expression evaluates to true?

Processing example 7.3

```
String str1 = "Dog";  
String str2 = "Cat";
```

str1.compareTo(str2) returns a positive integer as Dog (str1) comes after Cat (str2).

```
if(str1.compareTo(str2) < 0){  
    println(str1+" comes before "+str2 +" in the alphabet");  
}  
  
else if(str1.compareTo(str2) > 0){  
    println(str2 + " comes before "+ str1+" in the alphabet");  
}  
  
else{  
    println("The strings are identical");  
}
```

Processing example 7.4

```
String str1 = "cat";
```

```
String str2 = "Cat";
```

```
if(str1.compareTo(str2) < 0){
```

```
    println(str1+" comes before "+ str2 +" in the alphabet");
```

```
}
```

```
else if(str1.compareTo(str2) > 0){
```

```
    println(str2 + " comes before "+ str1+" in the alphabet");
```

```
}
```

```
else{
```

```
    println("The strings are identical");
```

```
}
```

What will be printed to the
console?

Which boolean expression
evaluates to true?

Processing example 7.4

```
String str1 = "cat";
String str2 = "Cat";
if(str1.compareTo(str2) < 0){
    println(str1+" comes before "+str2 +" in the alphabet");
}
else if(str1.compareTo(str2) > 0){
    println(str2 + " comes before "+str1+" in the alphabet");
}
else{
    println("The strings are identical");
}
```

str1.compareTo(str2) returns a positive integer as **cat (str1)** comes after **Cat (str2)** in the Unicode character map.

Processing example 7.5

```
String str1 = "Animal";
String str2 = "Cat";

if(str1.compareTo(str2) < 0){
    println(str1+" comes before "+str2 +" in the alphabet");
}
else if(str1.compareTo(str2) > 0){
    println(str2 + " comes before "+str1+" in the alphabet");
}
else{
    println("The strings are identical");
}
```

What will be printed to the console?
Which boolean expression evaluates to true?

Processing example 7.5

```
String str1 = "Animal";  
String str2 = "Cat";
```

str1.compareTo(str2) returns a negative integer as **Animal(str1)** comes before **Cat (str2)** in the Unicode character map.

```
if(str1.compareTo(str2) < 0){  
    println(str1+" comes before "+str2 +" in the alphabet");  
}  
  
else if(str1.compareTo(str2) > 0){  
    println(str2 + " comes before "+ str1+" in the alphabet");  
}  
else{  
    println("The strings are identical");  
}
```

Processing example 7.6

```
String str1 = "Cat";
String str2 = "Cat";

if(str1.compareTo(str2) < 0){
    println(str1+" comes before "+str2 +" in the alphabet");
}

else if(str1.compareTo(str2) > 0){
    println(str2 + " comes before "+str1+" in the alphabet");
}

else{
    println("The strings are identical");
}
```

What will be printed to the console?
Which boolean expression evaluates to true?

Processing example 7.6

```
String str1 = "Cat";  
String str2 = "Cat";
```

str1.compareTo(str2) returns 0
as Cat (str1) is identical to Cat
(str2).

```
if(str1.compareTo(str2) < 0){  
    println(str1+" comes before "+str2 +" in the alphabet");  
}  
else if(str1.compareTo(str2) > 0){  
    println(str2 + " comes before "+ str1+" in the alphabet");  
}  
else{  
    println("The strings are identical");  
}
```

Topics list

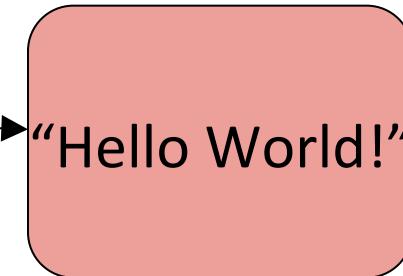
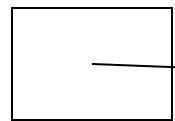
- Strings: index of characters
- String methods:
 - charAt(int index)
 - substring(int beginIndex, int endIndex)
 - compareTo(String anotherString)
- Recap: Primitive vs object
- String Identity versus equality
- null
- Escape Sequences

Recap: Object types e.g. String

- Strings are a sequence of characters enclosed by double quotes ("").
- String is an object type.
- The Java API provides information the String class and lists methods that can be used on Strings (
<https://docs.oracle.com/javase/8/docs/api/java/lang/String.html>).
- The most direct way to create a String is to write:
String greeting = "Hello world!";

Recap: Primitive types vs. object types

```
String greeting;
```



String is an object type. The greeting variable contains a reference to where the String is stored in memory.

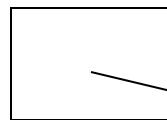
```
int i;
```

17

primitive
type

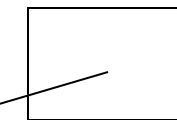
Recap: Primitive types vs. object types

String a;



“Hello World!”

String b;



b = a;

int a;



int b;



Topics list

- Strings: index of characters
- String methods:
 - `charAt(int index)`
 - `substring(int beginIndex, int endIndex)`
 - `compareTo(String anotherString)`
- Recap: Primitive vs object
- String Identity versus equality
- null
- Escape Sequences

String: Identity vs Equality (1)

```
if(input == "bye") {  
    ...  
}
```

tests identity
i.e. the
reference

```
if(input.equals("bye")) {  
    ...  
}
```

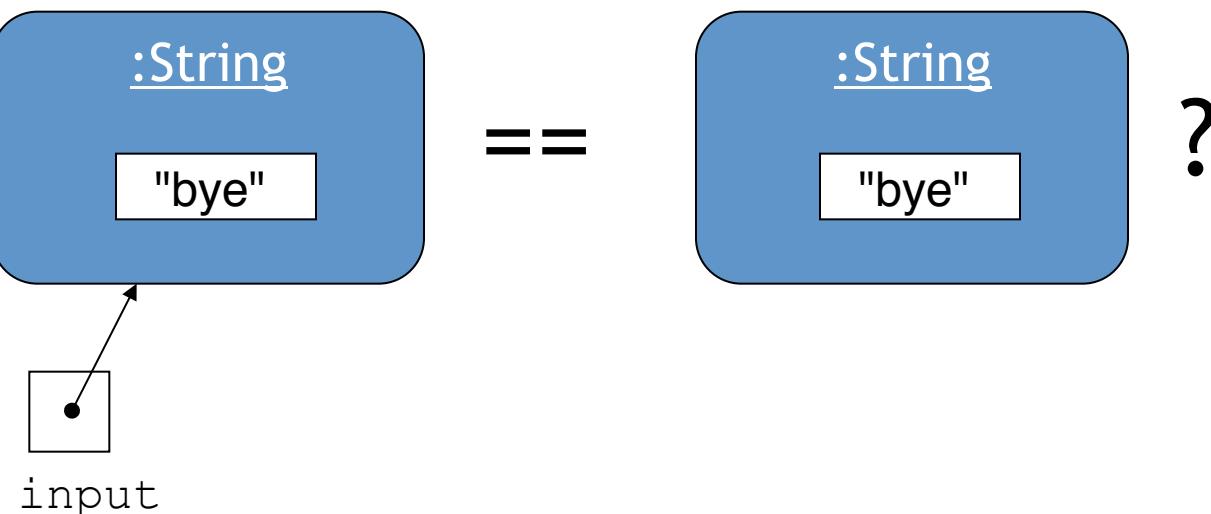
tests equality

Strings should always be compared
using the `.equals` method

String: Identity vs Equality (2)

```
String input = "bye";  
if(input == "bye") {  
    ...  
}
```

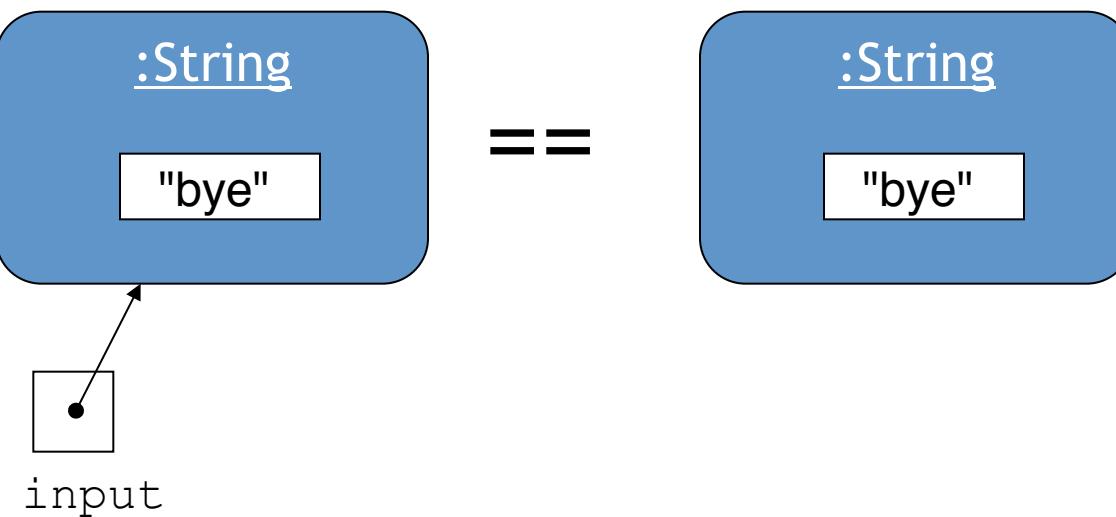
`==` tests identity



String: Identity vs Equality (2)

```
String input = "bye";  
if(input == "bye") {  
    ...  
}
```

`==` tests identity

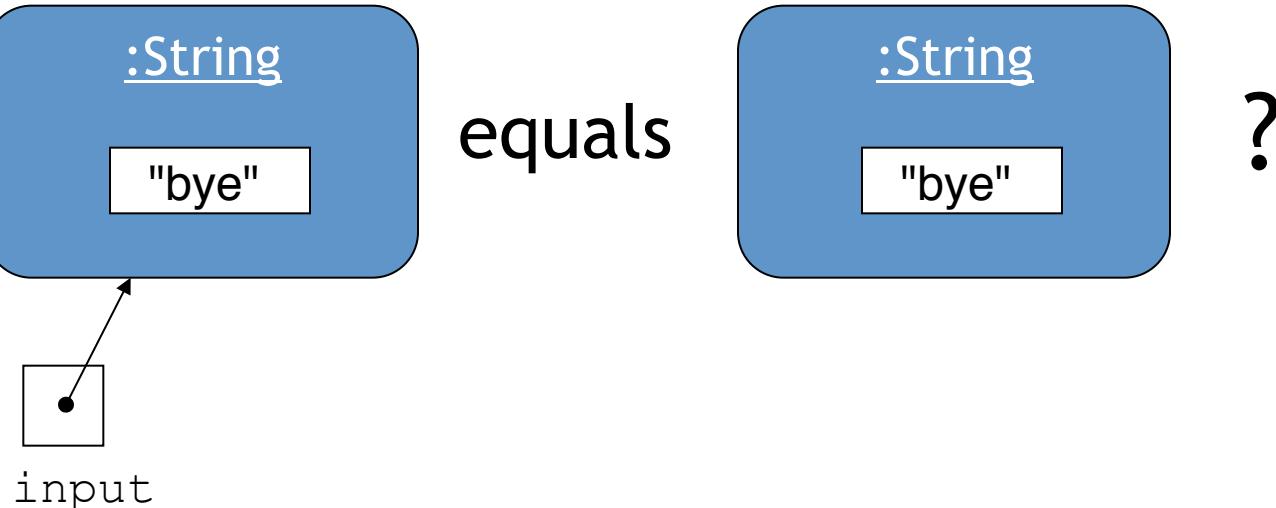


⇒ (may be) false!

String: Identity vs Equality (3)

```
String input = "bye";  
if(input.equals("bye")) {  
    ...  
}
```

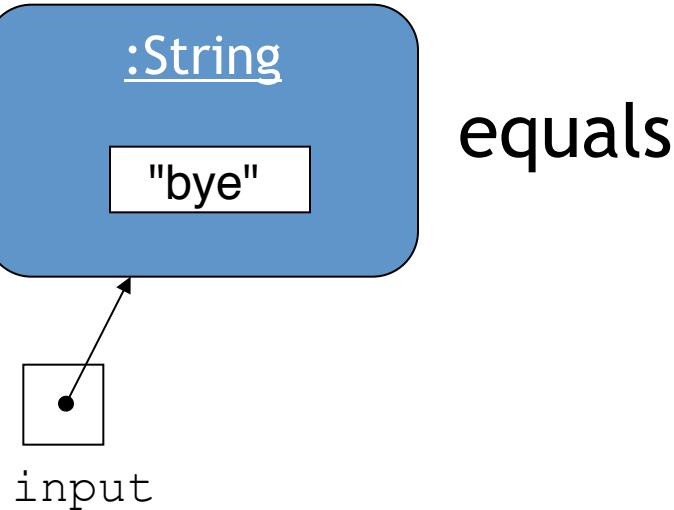
equals tests
equality



String: Identity vs Equality (3)

```
String input = "bye";  
if(input.equals("bye")) {  
    ...  
}
```

equals tests
equality



→ true!

Some common errors with Strings...

What's wrong here?

```
void anyMethod()
{
    String str1 = "a";
    String str2 = "b";

    if(str1 == str2)
    {
        println(str1+" is the same as
"+ str2);
    }
    else
    {
        println(str1+" is NOT same as
"+ str2);
    }
}
```

Strings need to use the .equals method

```
void anyMethod()
{
    String str1 = "a";
    String str2 = "b";

    if(str1 == str2)
    {
        println(str1+" is the same as
"+ str2);
    }
    else
    {
        println(str1+" is NOT same as
"+ str2);
    }
}
```

The code demonstrates string comparison. A red circle highlights the opening brace of the if block, likely to emphasize that the comparison `str1 == str2` is incorrect for strings. The correct way to compare strings is to use the `.equals` method.

What's wrong here?

```
public void anyMethod( )
{
    int num1 = 1;
    int num2 = 2;

    if(num1 = num2)
        println(num1+" is the same as "+
num2);
    else
        println(num1+" is NOT same as "+
num2);
}
```

You need two equals for equality

```
public void anyMethod( )
{
    int num1 = 1;
    int num2 = 2;

    if(num1 = num2)
        O println(num1+" is the same as "+
num2);
    else
        println(num1+" is NOT same as "+
num2);
}
```

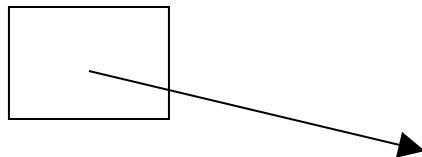
Topics list

- Strings: index of characters
- String methods:
 - `charAt(int index)`
 - `substring(int beginIndex, int endIndex)`
 - `compareTo(String anotherString)`
- Recap: Primitive vs object
- String Identity versus equality
- `null`
- Escape Sequences

null

- **null** is a special value in Java.
- All object variables are initialised to **null**.
- null means that the object variable does not have a reference e.g. str2 below.

String str1;



String str2;



null

- **null** is a special value in Java
- All object variables are initialised to **null**.
- You can assign and test for **null**:

```
private NumberDisplay hours;
```

```
if (hours == null) { ... }
```

```
hours = null;
```

Topics list

- Strings: index of characters
- String methods:
 - `charAt(int index)`
 - `substring(int beginIndex, int endIndex)`
 - `compareTo(String anotherString)`
- Recap: Primitive vs object
- String Identity versus equality
- null
- Escape Sequences

Escape sequences

- When a String is printed, certain single characters that follow a backslash (\) have special meaning...
- ...and the compiler interprets them accordingly.

Java escape sequences

Escape Sequence	Description
\t	Insert a tab in the text at this point.
\b	Insert a backspace in the text at this point.
\n	Insert a newline in the text at this point.
\r	Insert a carriage return in the text at this point.
\f	Insert a formfeed in the text at this point.
\'	Insert a single quote character in the text at this point.
\"	Insert a double quote character in the text at this point.
\\\	Insert a backslash character in the text at this point.

Examples of escape sequences

```
print("Java\n");
```

is the exact same as:

```
println("Java");
```

```
println("    Java");
```

is similar to:

```
println("\tJava");
```

Questions?





Except where otherwise noted, this content is licensed under a Creative Commons Attribution-NonCommercial 3.0 License.

For more information, please see <http://creativecommons.org/licenses/by-nc/3.0/>



Waterford Institute of Technology
INSTITIÚID TEICNEOLAÍOCHTA PHORT LÁIRGE

Department of Computing and Mathematics
<http://www.wit.ie/>