Strings

Strings and their methods

Produced by: Dr. Siobhán Drohan

Waterford Institute *of* Technology
INSTITIÚID TEICNEOLAÍOCHTA PHORT LÁIRGE

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

Topics list

- Primitive Types: char
- Object Types: String
- Primitive vs Object Types
- Strings and Java API
- Method calls (internal, external, dot notation)
- Escape Sequences
- Strings and methods:
  - charAt
  - substring
  - length
  - toUpperCase
  - toLowerCase
  - trim
  - compareTo
  - equals

Primitive Types

- Java programming language supports eight primitive data types.

- The char data type stores one single character which is delimited by single quotes(‘) e.g.

    char letter = ‘a’;

Data Type	Default Value
byte	0
short	0
int	0
long	0L
float	0.0f
double	0.0d
char	'\u0000'
boolean	false

Primitive Types: char

```
// VALID USE
char letter  = ‘n’;        //Assign ‘n’ to the letter variable
char letter = ‘N’;         //Assign ‘N’ to the letter variable


// INVALID USE
char letter = n;           //ERROR – no single quotes around n.
char letter = “n”;         //ERROR – double quotes around n.
char letter = “not”;       //ERROR – char can only hold one character.
```

Source: Reas & Fry (2014)

Primitive Types: char

- char is a 16-bit Unicode character.
- Values range:
  - from '\u0000‘ (or 0)
  - to '\uffff' (or 65,535)
- For example:
  - ‘A’ is ‘\u0041’
  - ‘a’ is ‘\u0061’

http://en.wikipedia.org/wiki/List_of_Unicode_characters

Topics list

- Primitive Types: char
- Object Types: String
- Primitive vs Object Types
- Strings and Java API
- Method calls (internal, external, dot notation)
- Escape Sequences
- Strings and methods:
  - charAt
  - substring
  - length
  - toUpperCase
  - toLowerCase
  - trim
  - compareTo
  - equals

Object types e.g. String

- Strings, which are widely used in Java, are a sequence of characters enclosed by double quotes (“).
- In Java, a String is an object type.
- The Java platform provides the String class to create and manipulate strings.
- The most direct way to create a String is to write:

```
String greeting = "Hello world!";
```

http://www.tutorialspoint.com/java/pdf/java_strings.pdf

Object types - String

```
// VALID USE
String str = “I am a sentence”;  //Assigns the full sentence to str variable.
String word = “dog”;       //Assigns the word “dog” to the word variable.
String letter = “A”;       //Assigns the letter “A” to the letter variable.

// INVALID USE
String letter = n;         //ERROR – no double quotes around n.
String letter = ‘n’;       //ERROR – single quotes around n; use double.
string letter = “n”;       //ERROR – String should have a capital S.
```

Source: Reas & Fry (2014)

Topics list

- Primitive Types: char
- Object Types: String
- Primitive vs Object Types
- Strings and Java API
- Method calls (internal, external, dot notation)
- Escape Sequences
- Strings and methods:
  - charAt
  - substring
  - length
  - toUpperCase
  - toLowerCase
  - trim
  - compareTo
  - equals

Primitive vs. Object Types

Primitive type

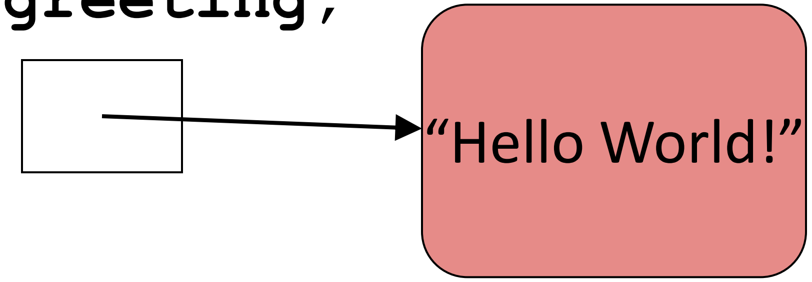
```
int i;
```

$\boxed{17}$

With primitive type variables (e.g. int, float, char, etc) the value of the variable is stored in the memory location assigned to the variable.

Primitive vs. Object Types

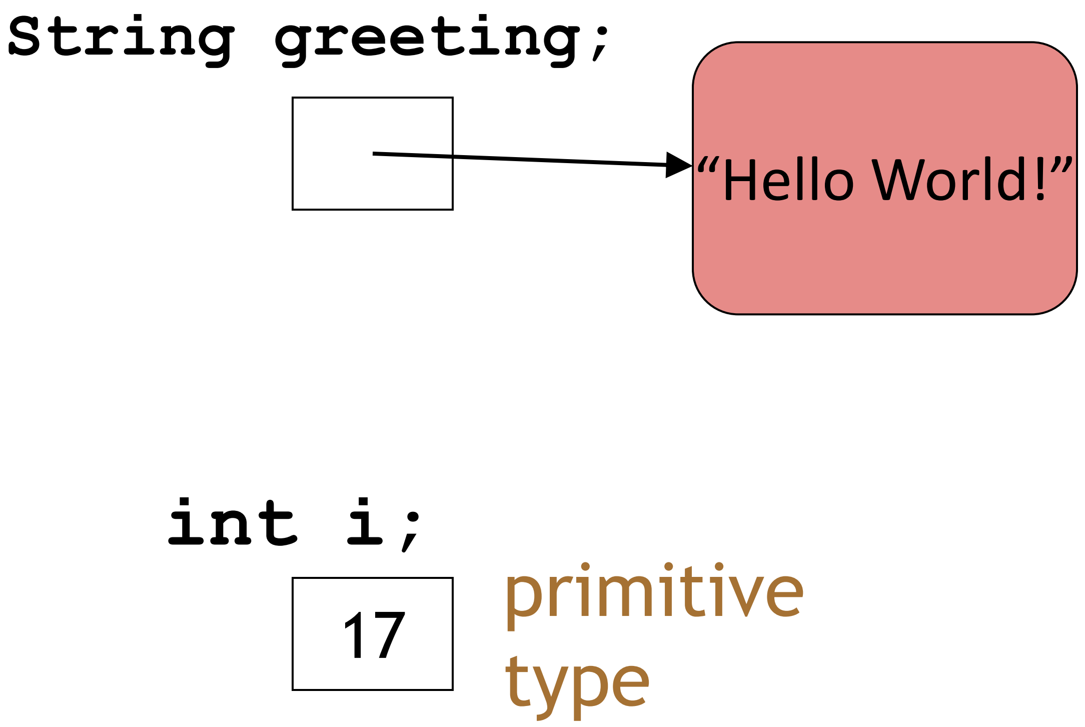
```
String greeting;
```



With object types, the variable holds the memory address of where the object is located – it does not store the values inside the object.

This memory address is called a **reference** to the object.

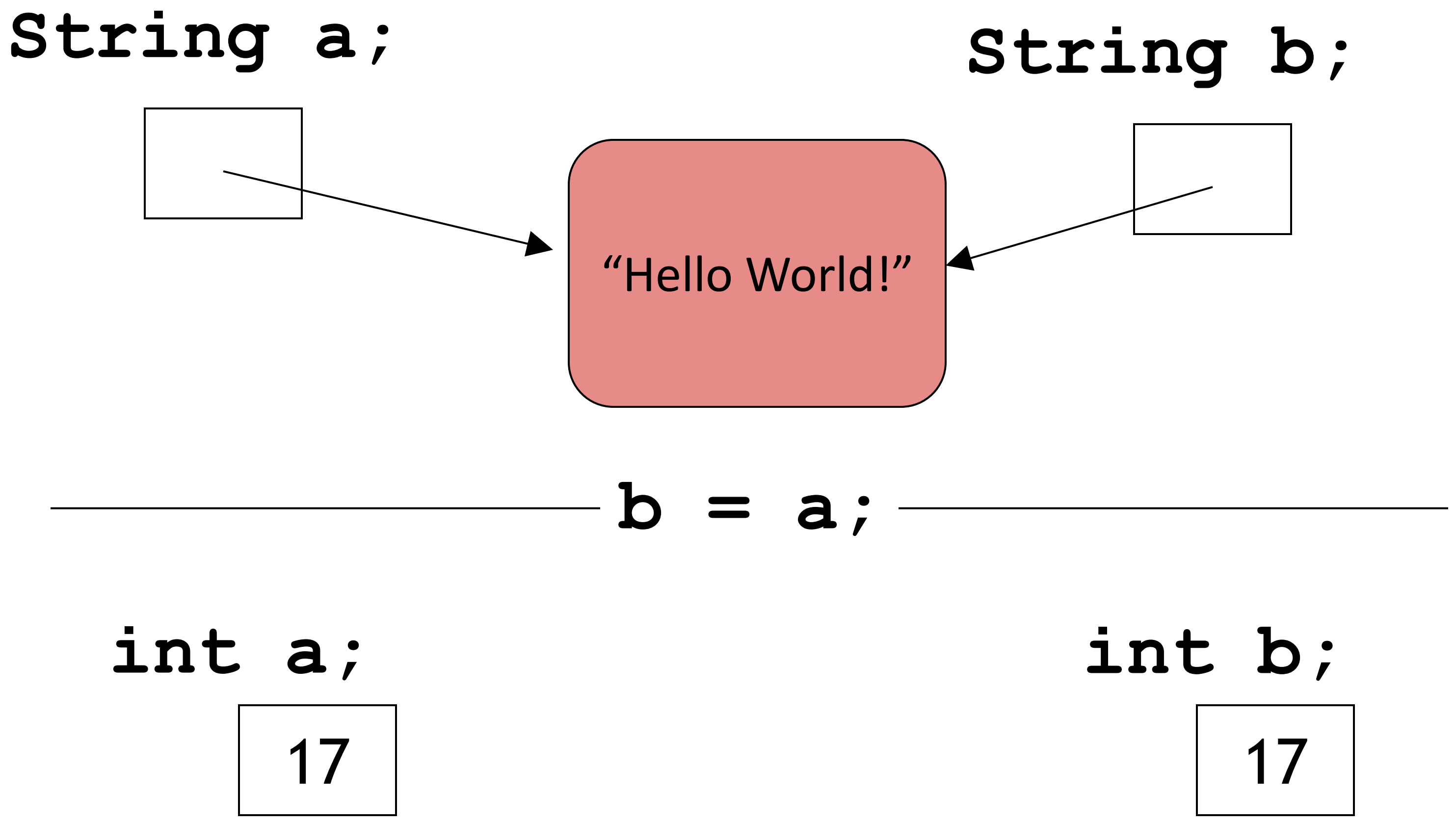
Primitive vs. Object Types



String is an object type.

The **greeting** variable contains a reference to where the String is stored in memory.

Primitive vs. Object Types



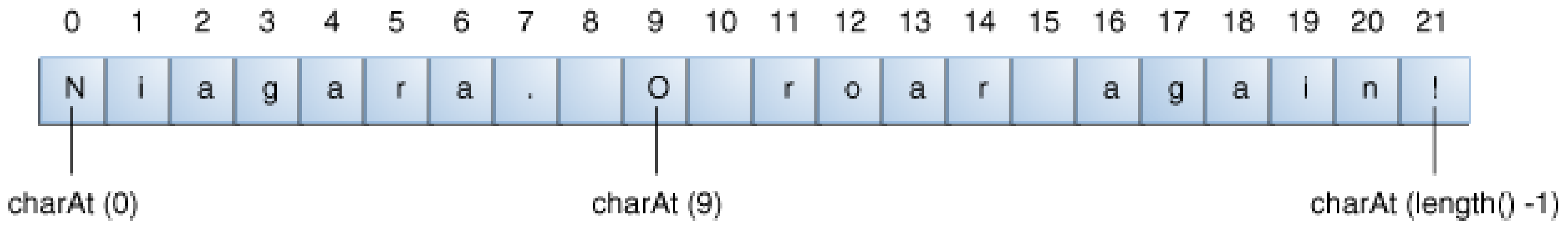
Topics list

- Primitive Types: char
- Object Types: String
- Primitive vs Object Types
- Strings and Java API
- Method calls (internal, external, dot notation)
- Escape Sequences

- Strings and methods:
  - charAt
  - substring
  - length
  - toUpperCase
  - toLowerCase
  - trim
  - compareTo
  - equals

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.



https://docs.oracle.com/javase/tutorial/java/data/manipstrings.html

Strings are objects

- Variables created with the String data type are called objects.

- Objects are software structures that combine variables with methods that operate on those variables e.g.
  - every String object has a built-in method that can capitalise its letters.

Strings and Java's API

- This link is to Java's Application Programming Interface (API).

  https://docs.oracle.com/javase/8/docs/api/index.html?overview-summary.html

- At the moment, we are interested in finding out more information on String, particularly its methods:

  https://docs.oracle.com/javase/8/docs/api/java/lang/String.html

Topics list

- Primitive Types: char
- Object Types: String
- Primitive vs Object Types
- Strings and Java API
- Method calls (internal, external, dot notation)
- Escape Sequences

- Strings and methods:
  - charAt
  - substring
  - length
  - toUpperCase
  - toLowerCase
  - trim
  - compareTo
  - equals

External method calls

- Say we want to check the length of this String:

  String name = “Joe Soap”;

- Looking at the String API, we can see this method:

ReturnType	Method	Description
int	length()	Returns the length of this string.

- A call to a method of another object is called an external method call.

External method calls

- External method calls have the syntax:

  object.methodname (parameter-list)

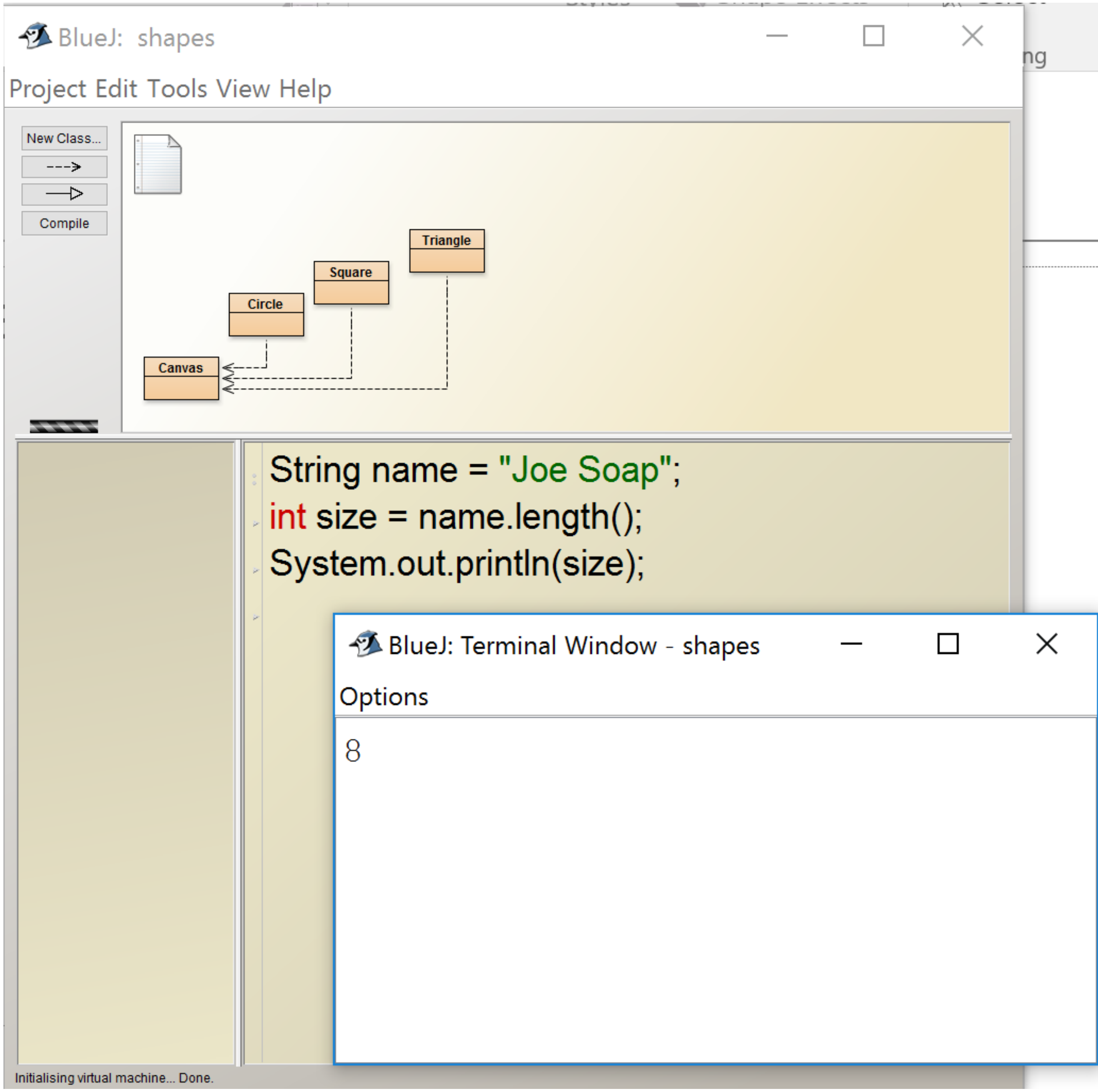
- To find out the length of this String:

```
String name = “Joe Soap”;
```

- We make the following external method call:

```
name.length();
```

External method call


BlueJ: shapes
Project Edit Tools View Help
New Class...
Compile
Triangle
Square
Circle
Canvas
BlueJ: Terminal Window - shapes
Options
Initialising virtual machine... Done.

```
String name = "Joe Soap";
int size = name.length();
System.out.println(size);
```

```
8
```

Dot Notation

- Methods can call methods of other objects using dot notation.
- This syntax is known as dot notation:

  object.methodname (parameter-list)
- It consists of:
  - An object
  - A dot
  - A method name
  - The parameters for the method

Topics list

- Primitive Types: char
- Object Types: String
- Primitive vs Object Types
- Strings and Java API
- Method calls (internal, external, dot notation)
- Escape Sequences
- Strings and methods:
  - charAt
  - substring
  - length
  - toUpperCase
  - toLowerCase
  - trim
  - compareTo
  - equals

Java Escape Sequences

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

http://docs.oracle.com/javase/tutorial/java/data/characters.html

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.

http://docs.oracle.com/javase/tutorial/java/data/characters.html

Java Escape Sequences - examples

```
System.out.print(“Java\n”);
```

is the exact same as:

```
System.out.println(“Java”);
```

```
System.out.println(“	Java”);
```

is similar to:

```
System.out.println(“\tJava”);
```

http://docs.oracle.com/javase/tutorial/java/data/characters.html

Topics list

- Primitive Types: char
- Object Types: String
- Primitive vs Object Types
- Strings and Java API
- Method calls (internal, external, dot notation)
- Escape Sequences

- Strings and methods:
  - charAt
  - substring
  - length
  - toUpperCase
  - toLowerCase
  - trim
  - compareTo
  - equals

Strings and some API methods

Return Type	Method Name	Description
char	charAt(int index)	Returns the char value at the specified index.
String	substring(int beginIndex, int endIndex)	Returns a string that is a substring of this string.
int	length()	Returns the length of this string.
String	toUpperCase()	Converts all of the characters in this String to upper case.
String	toLowerCase()	Converts all of the characters in this String to lower case.
String	trim()	Returns a string whose value is this string, with any leading and trailing whitespace removed.
int	compareTo(String anotherString)	Compares two strings lexicographically (i.e. unicode ordering).
boolean	equals (Object anObject)	Compares this string to the specified object.

https://docs.oracle.com/javase/8/docs/api/java/lang/String.html

Topics list

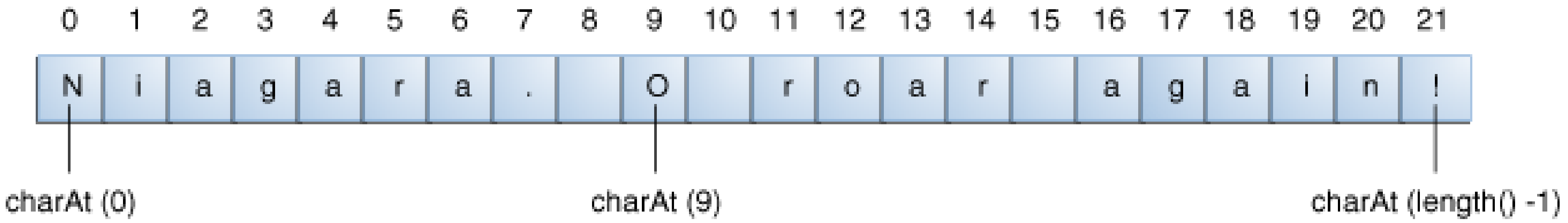
Return type	char
Method	charAt(int index)
Description	Returns the char value at the specified index.

- Strings and methods:
  - charAt
  - substring
  - length
  - toUpperCase
  - toLowerCase
  - trim
  - compareTo
  - equals

String methods: charAt(int index)

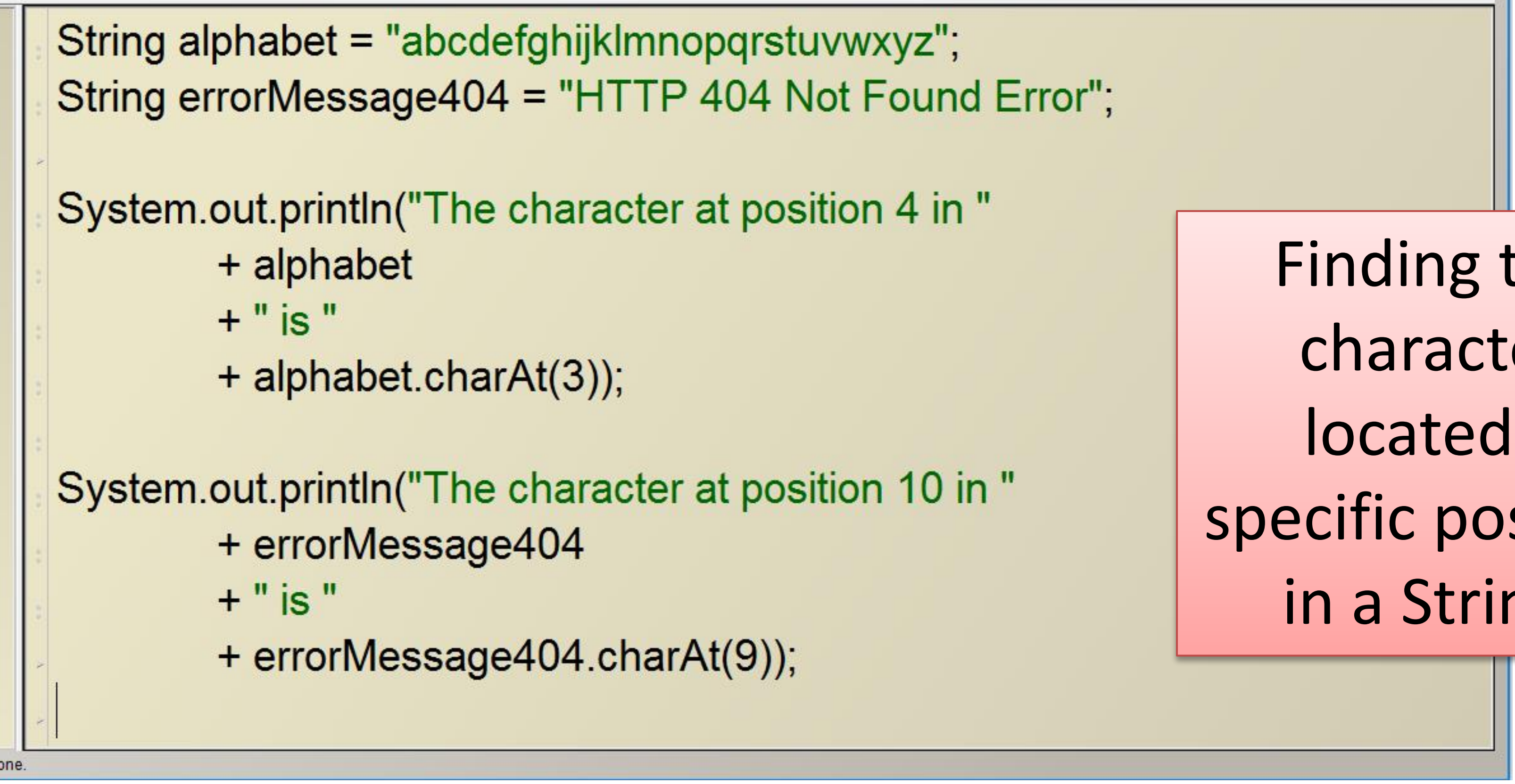
- The following code gets the character at index 9 in a String:

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



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

https://docs.oracle.com/javase/tutorial/java/data/manipstrings.html

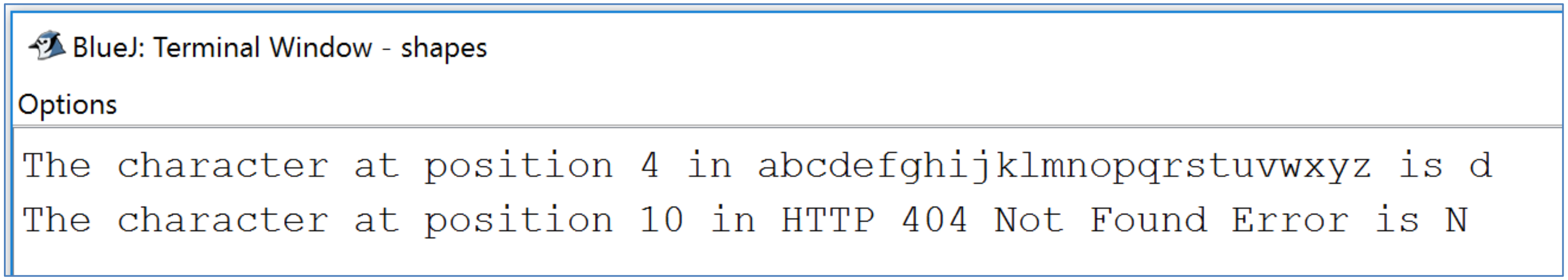


```
String alphabet = "abcdefghijklmnopqrstuvwxyz";
String errorMessage404 = "HTTP 404 Not Found Error";

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

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

Finding the character located a specific position in a String.



BlueJ: Terminal Window - shapes

Options

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

Topics list

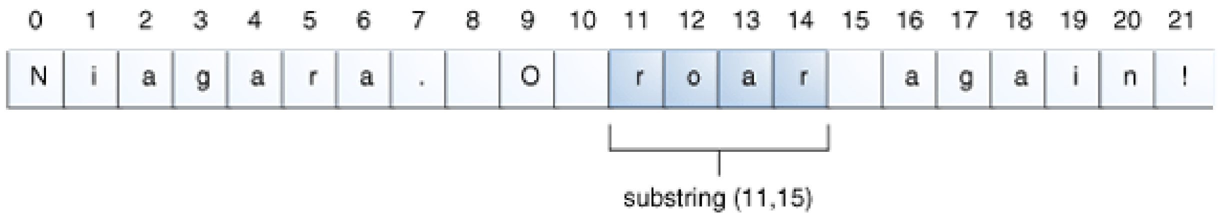
Return type	String
Method	substring(int beginIndex, int endIndex)
Description	Returns a string that is a substring of this string.

- Strings and methods:
  - charAt
  - substring
  - length
  - toUpperCase
  - toLowerCase
  - trim
  - compareTo
  - equals

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.

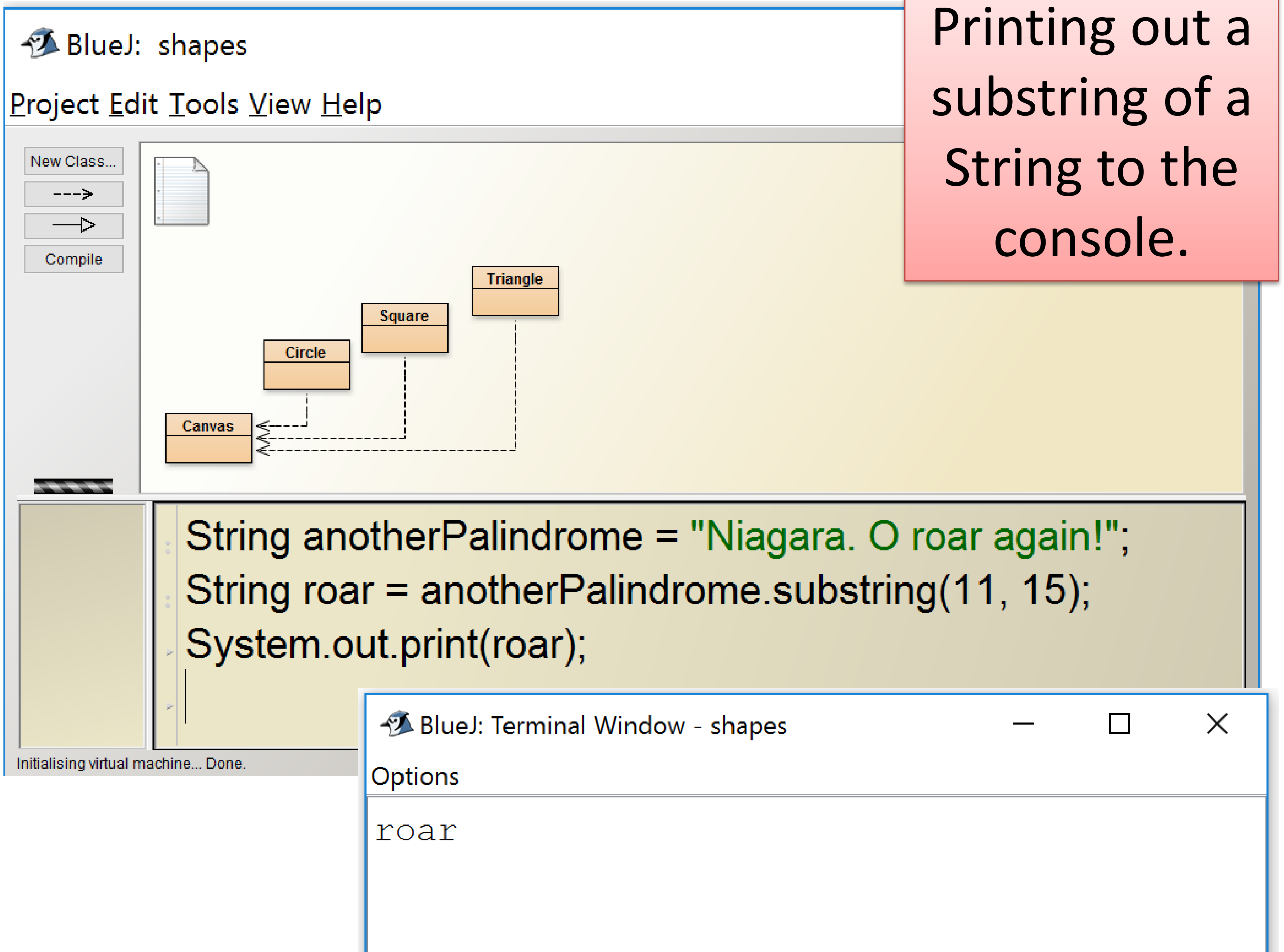
```
String anotherPalindrome = "Niagara. O roar again!";
String roar = anotherPalindrome.substring(11, 15);
```



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

https://docs.oracle.com/javase/tutorial/java/data/manipstrings.html

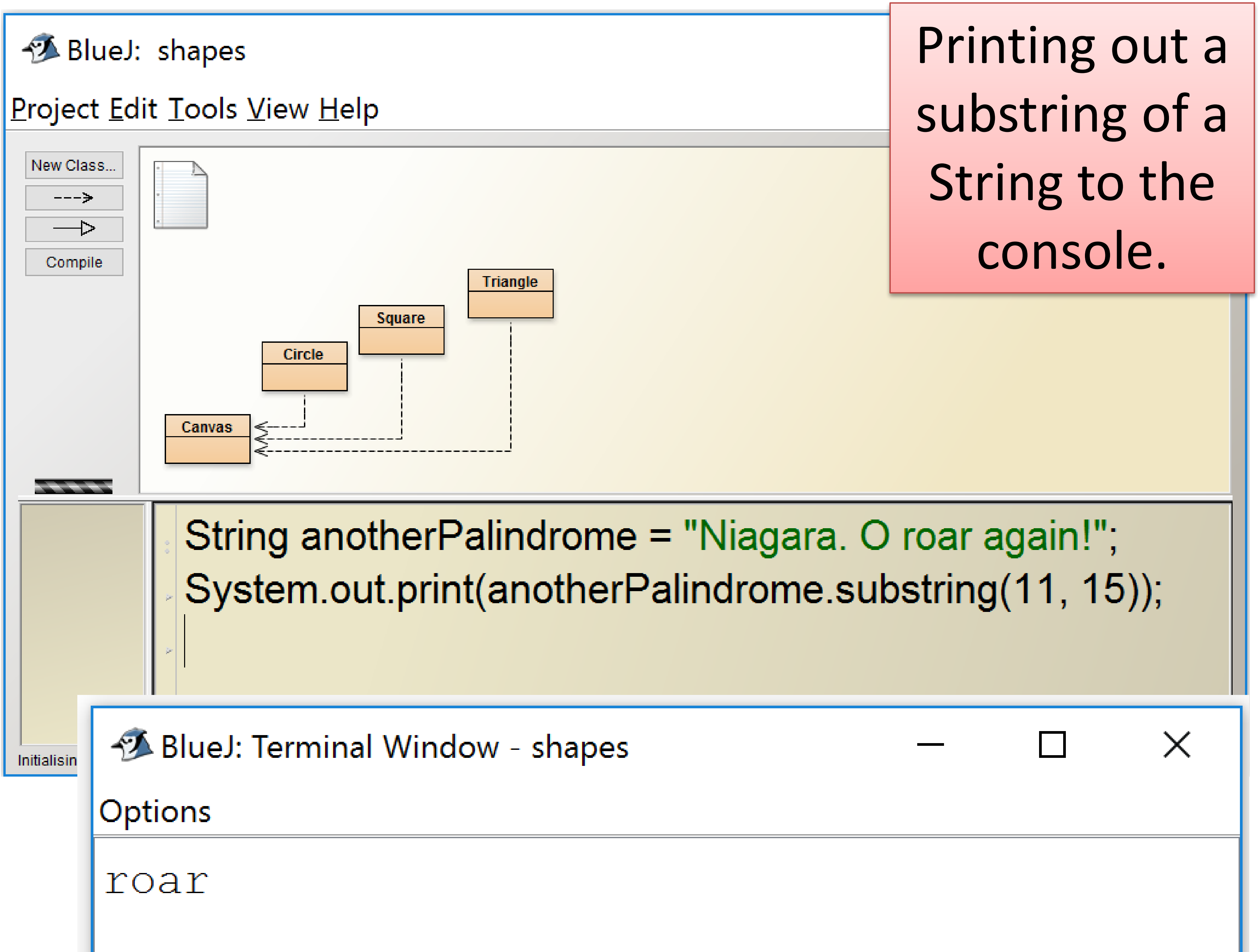
Printing out a substring of a String to the console.


BlueJ: shapes
Project Edit Tools View Help
New Class...
Compile
Triangle
Square
Circle
Canvas
Initialising virtual machine... Done.
BlueJ: Terminal Window - shapes
Options

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

```
roar
```

Printing out a substring of a String to the console.



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

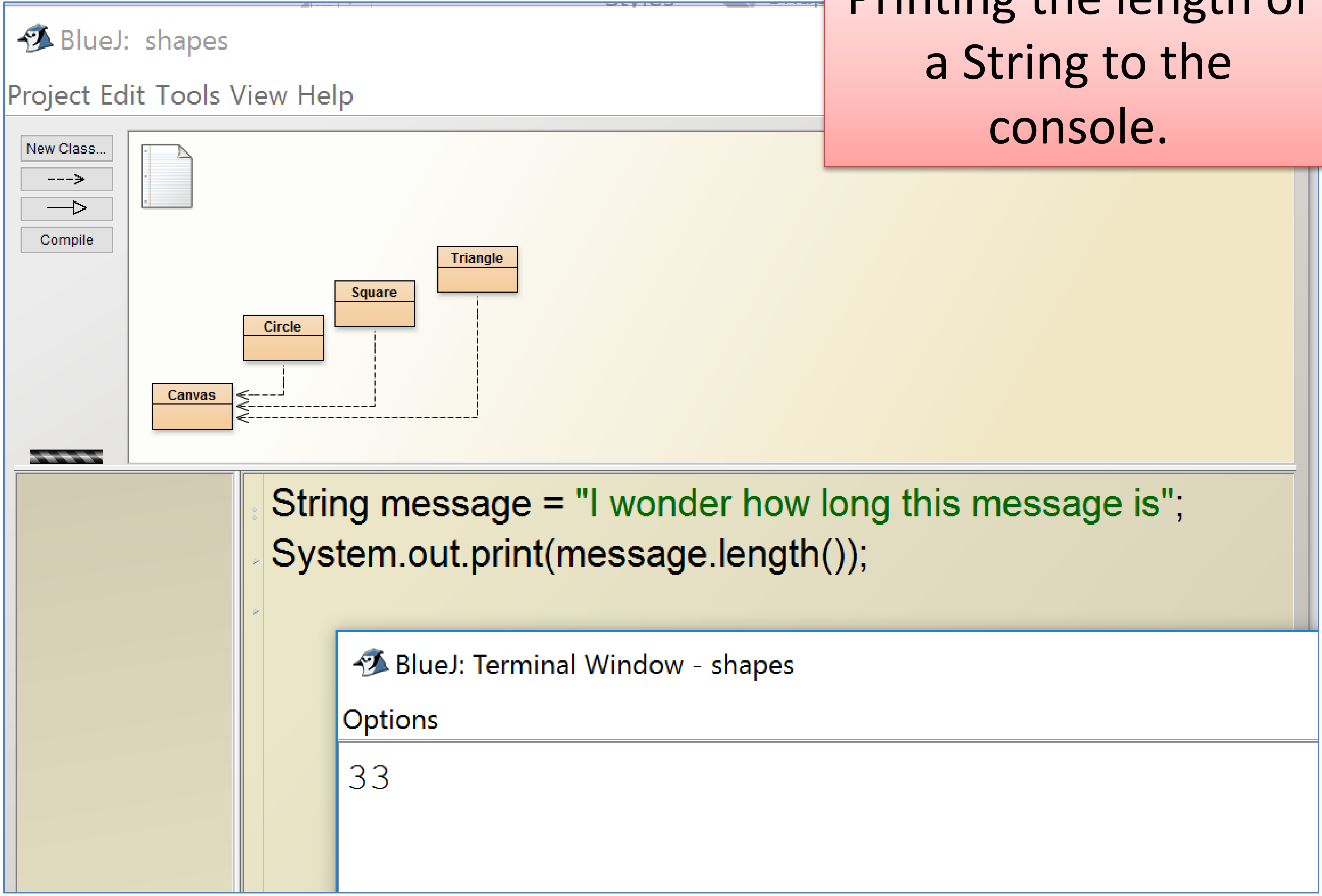
```
roar
```

Topics list

Return type	int
Method	length()
Description	Returns the length of this string.

- Strings and methods:
  - charAt
  - substring
  - length
  - toUpperCase
  - toLowerCase
  - trim
  - compareTo
  - equals

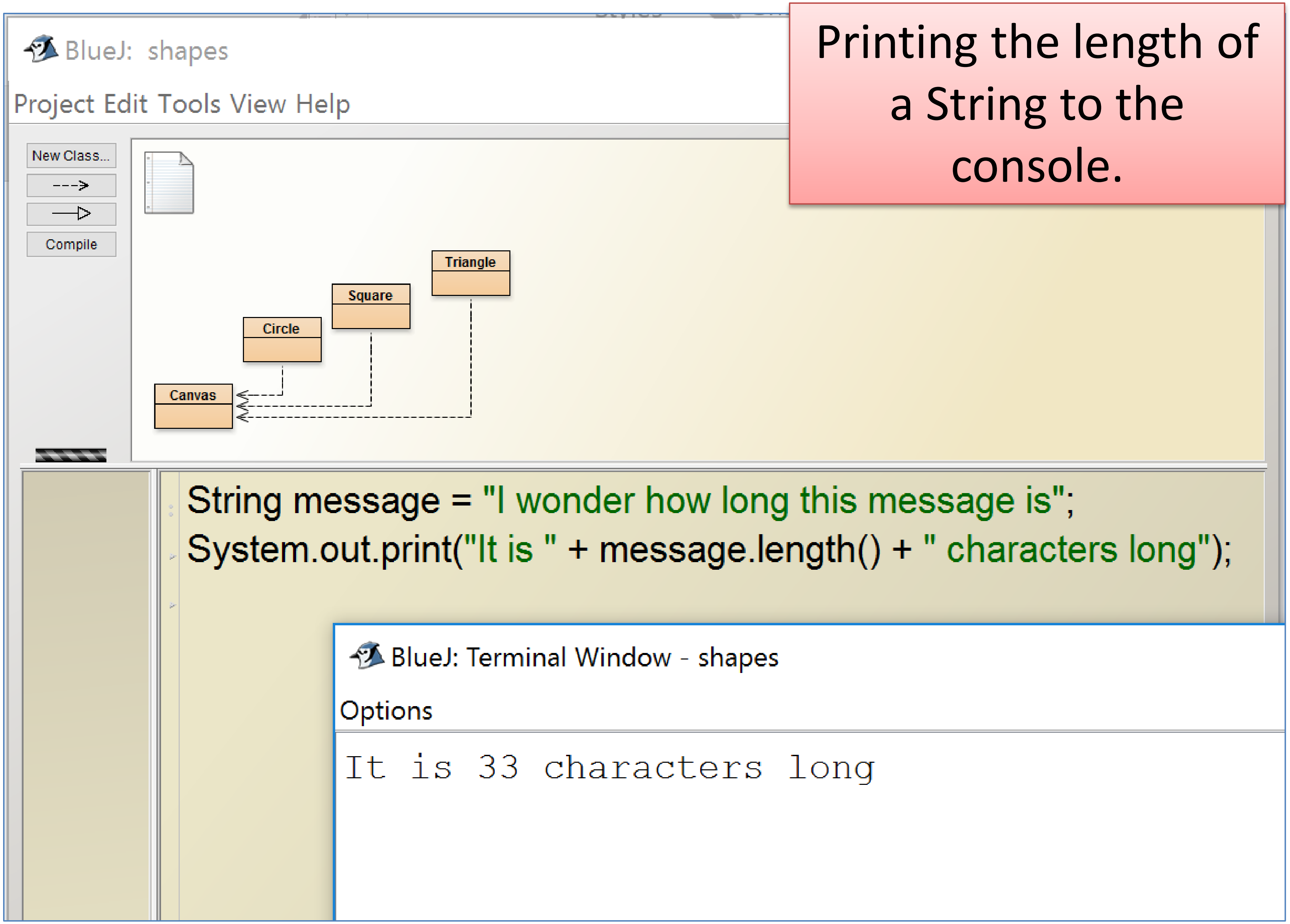
Printing the length of a String to the console.



```java
String message = "I wonder how long this message is";
System.out.print(message.length());
```

```
33
```

Printing the length of a String to the console.



```java
String message = "I wonder how long this message is";
System.out.print("It is " + message.length() + " characters long");
```

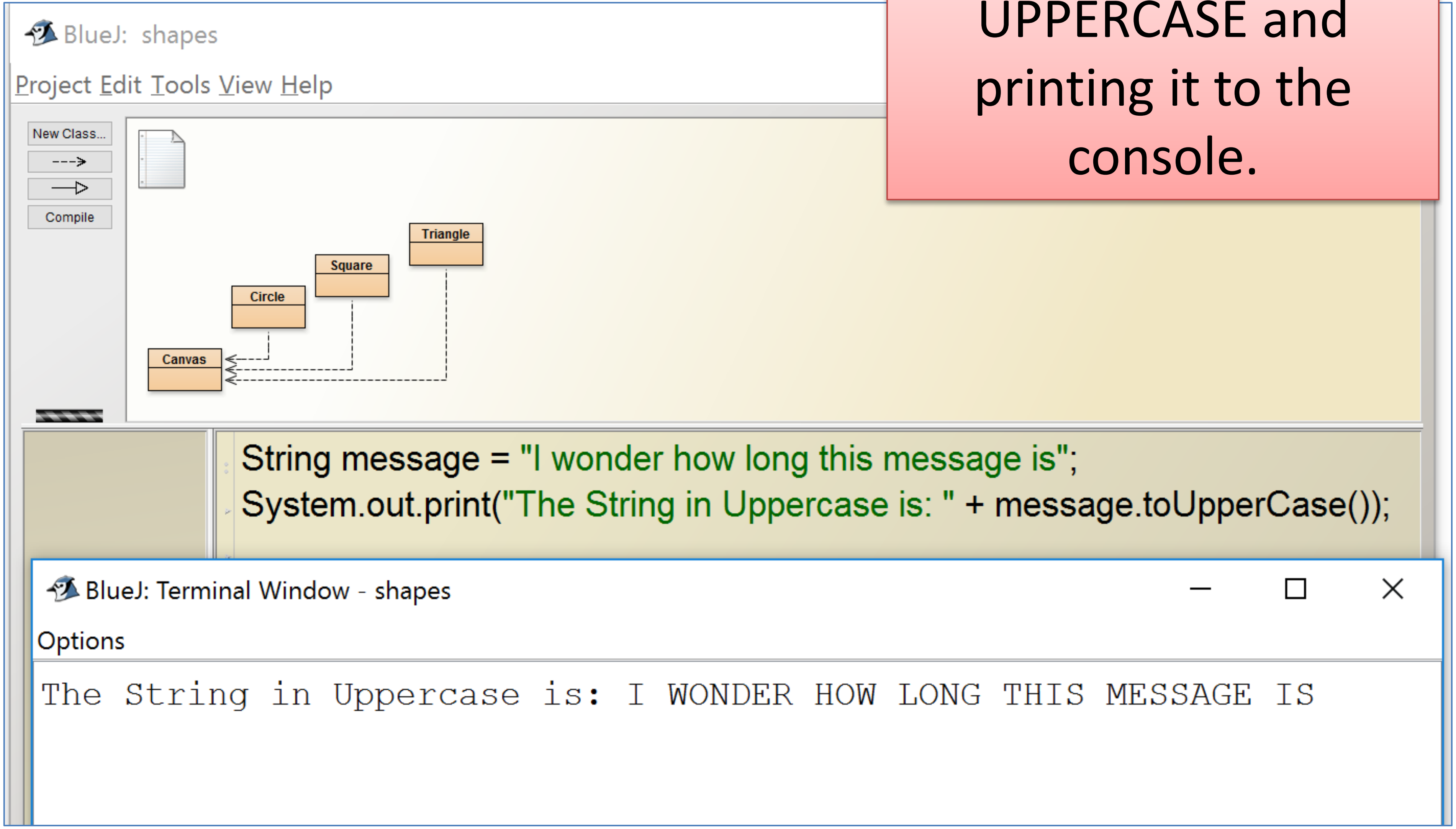
```
It is 33 characters long
```

Topics list

Return type	String
Method	toUpperCase()
Description	Converts all of the characters in this String to upper case.

- Strings and methods:
  - charAt
  - substring
  - length
  - toUpperCase
  - toLowerCase
  - trim
  - compareTo
  - equals

Converting a String to UPPERCASE and printing it to the console.



```
String message = "I wonder how long this message is";
System.out.print("The String in Uppercase is: " + message.toUpperCase());
```

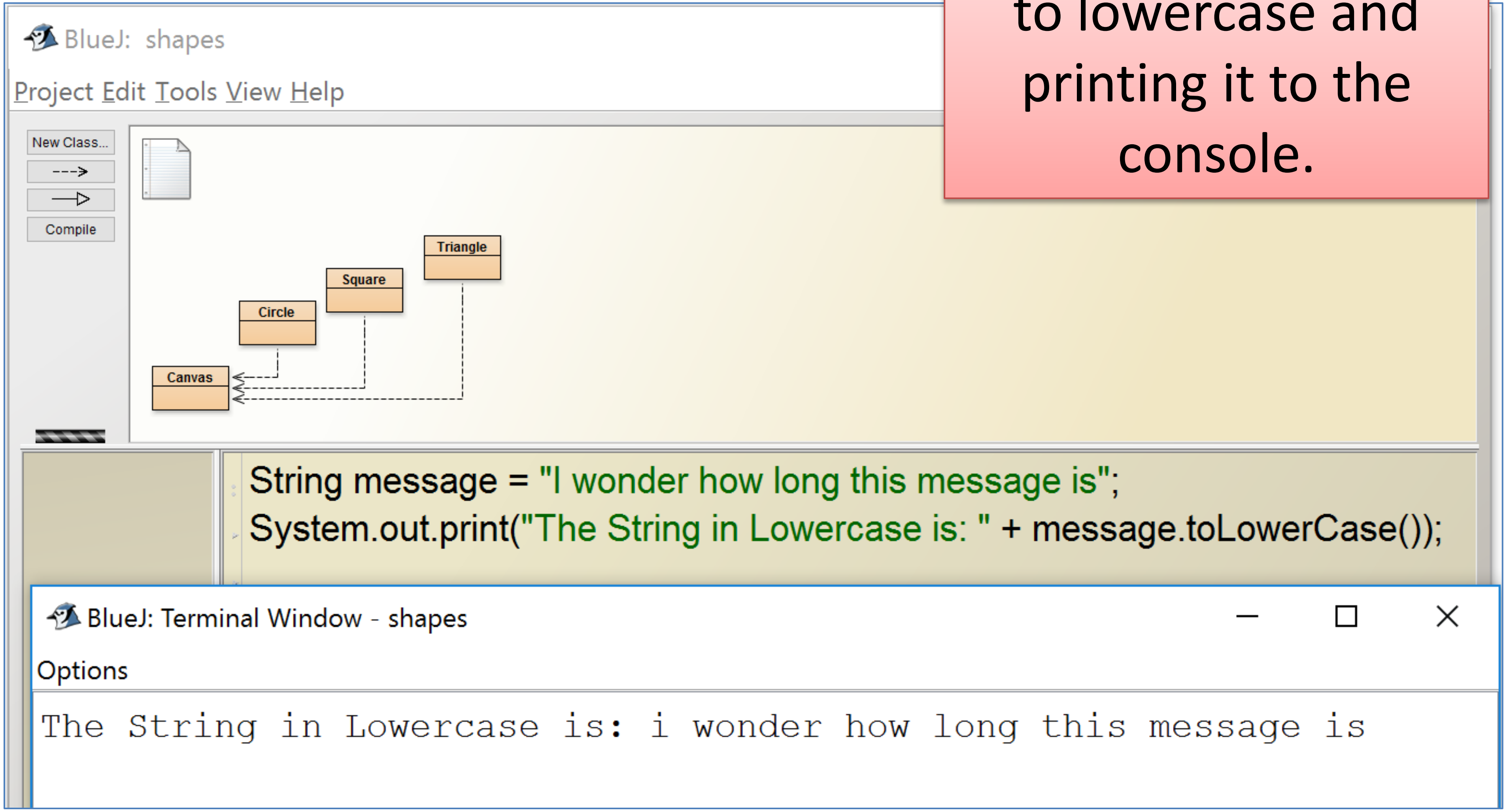
```
The String in Uppercase is: I WONDER HOW LONG THIS MESSAGE IS
```

Topics list

Return type	String
Method	toLowerCase()
Description	Converts all of the characters in this String to lower case.

- Strings and methods:
  - charAt
  - substring
  - length
  - toUpperCase
  - toLowerCase
  - trim
  - compareTo
  - equals

Converting a String to lowercase and printing it to the console.



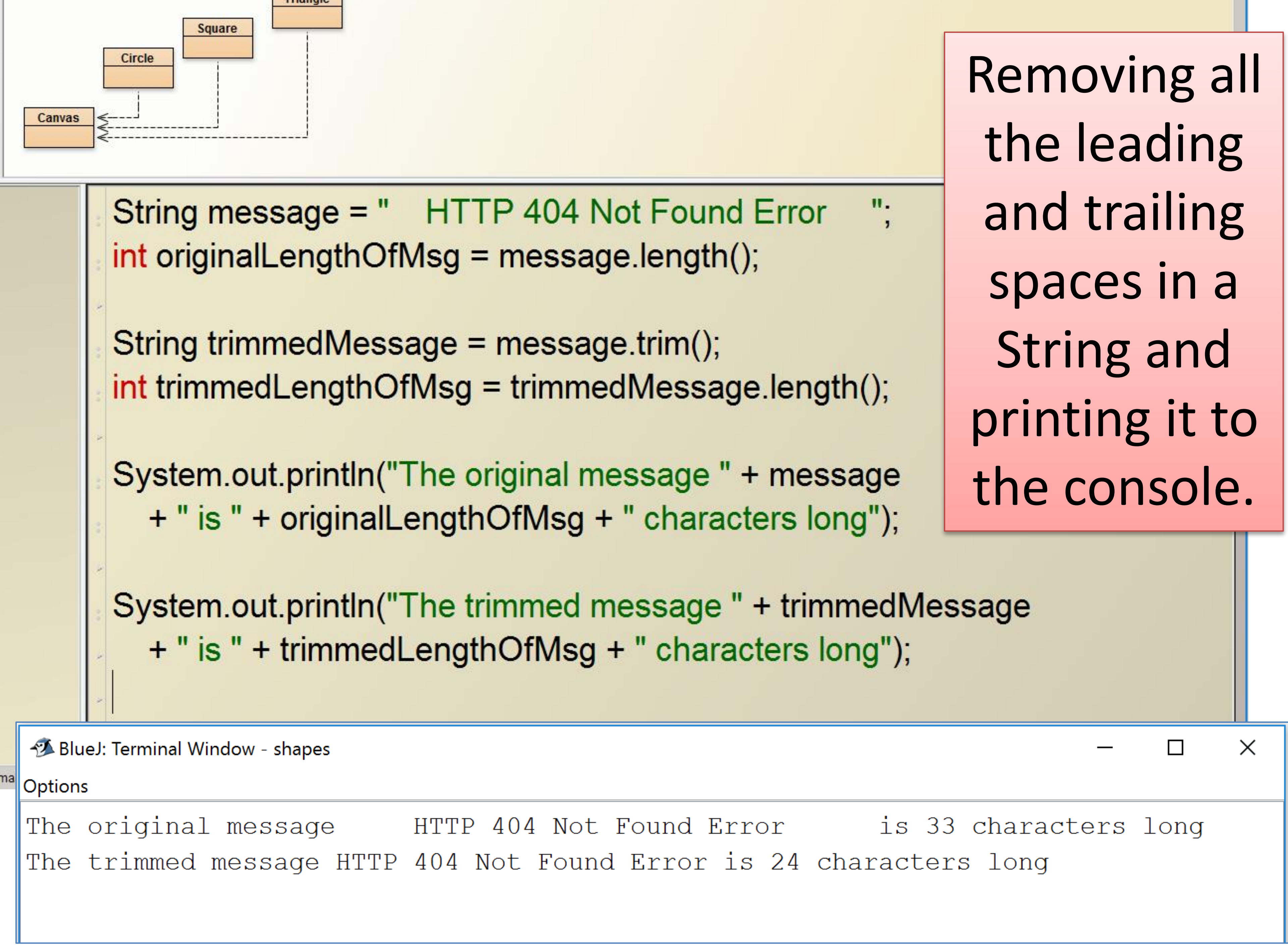
```
String message = "I wonder how long this message is";
System.out.print("The String in Lowercase is: " + message.toLowerCase());
```

```
The String in Lowercase is: i wonder how long this message is
```

Topics list

Return type	String
Method	trim()
Description	Returns a string whose value is this string, with any leading and trailing whitespace removed.

- Strings and methods:
  - charAt
  - substring
  - length
  - toUpperCase
  - toLowerCase
  - trim
  - compareTo
  - equals



```
String message = "    HTTP 404 Not Found Error     ";
int originalLengthOfMsg = message.length();

String trimmedMessage = message.trim();
int trimmedLengthOfMsg = trimmedMessage.length();

System.out.println("The original message " + message
    + " is " + originalLengthOfMsg + " characters long");

System.out.println("The trimmed message " + trimmedMessage
    + " is " + trimmedLengthOfMsg + " characters long");
```

```
The original message     HTTP 404 Not Found Error      is 33 characters long
The trimmed message HTTP 404 Not Found Error is 24 characters long
```

Removing all the leading and trailing spaces in a String and printing it to the console.

Topics list

Return type	int
Method	compareTo(String anotherString)
Description	Compares two strings lexicographically (i.e. unicode ordering).

- Strings and methods:
  - charAt
  - substring
  - length
  - toUpperCase
  - toLowerCase
  - trim
  - compareTo
  - equals

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.

https://docs.oracle.com/javase/tutorial/java/data/comparestrings.html

compareTo: Example 1

What will be printed to the console?

Which boolean expression evaluates to true?

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

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

compareTo: Example 1

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

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

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

compareTo: Example 2

What will be printed to the console?

Which boolean expression evaluates to true?

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

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

compareTo: Example 2

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

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

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

compareTo: Example 3

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

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

What will be printed to the console?

Which boolean expression evaluates to true?

compareTo: Example 3

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

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

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

compareTo: Example 4

What will be printed to the console?

Which boolean expression evaluates to true?

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

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

compareTo: Example 4

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

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

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

Topics list

Return type	boolean
Method	equals (Object anObject)
Description	Compares this string to the specified object.

- Strings and methods:
  - charAt
  - substring
  - length
  - toUpperCase
  - toLowerCase
  - trim
  - compareTo
  - equals

String: Identity vs Equality (1)

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

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

tests identity i.e. the reference

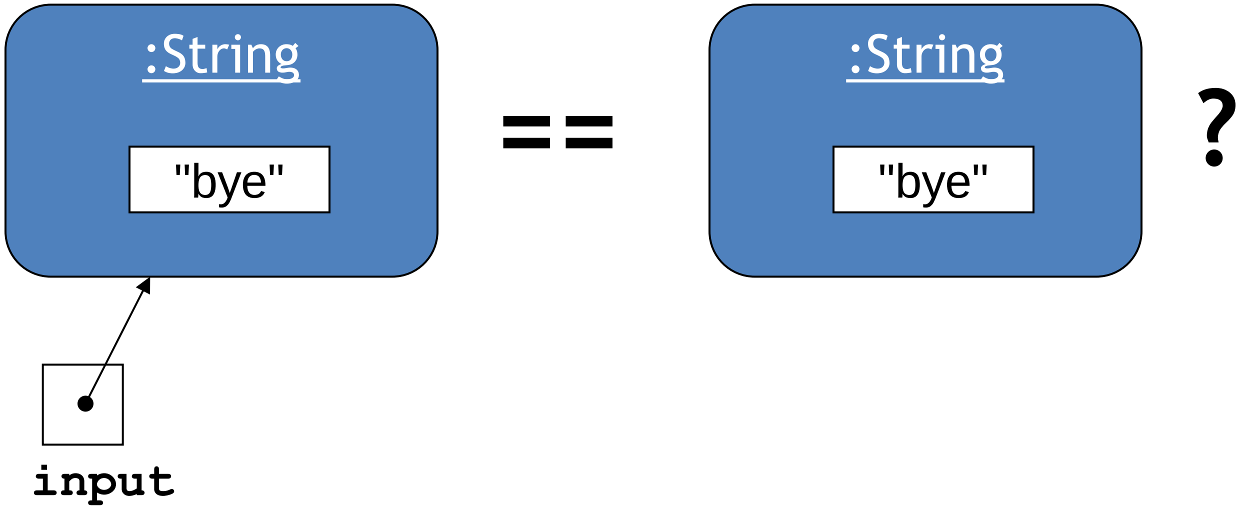
tests equality

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

String: Identity vs Equality (2)

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

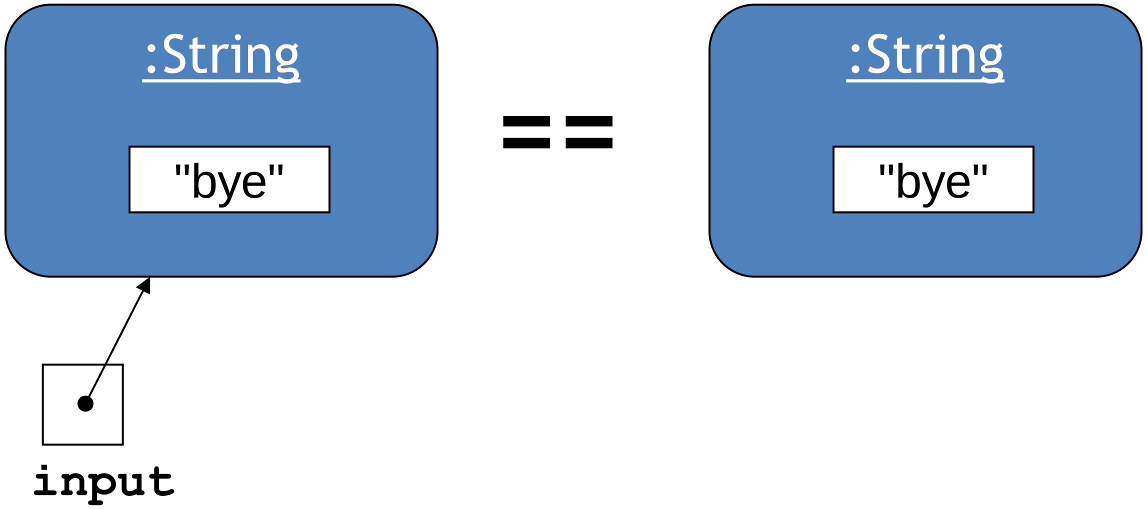
== tests identity



String: Identity vs Equality (2)

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

== tests identity

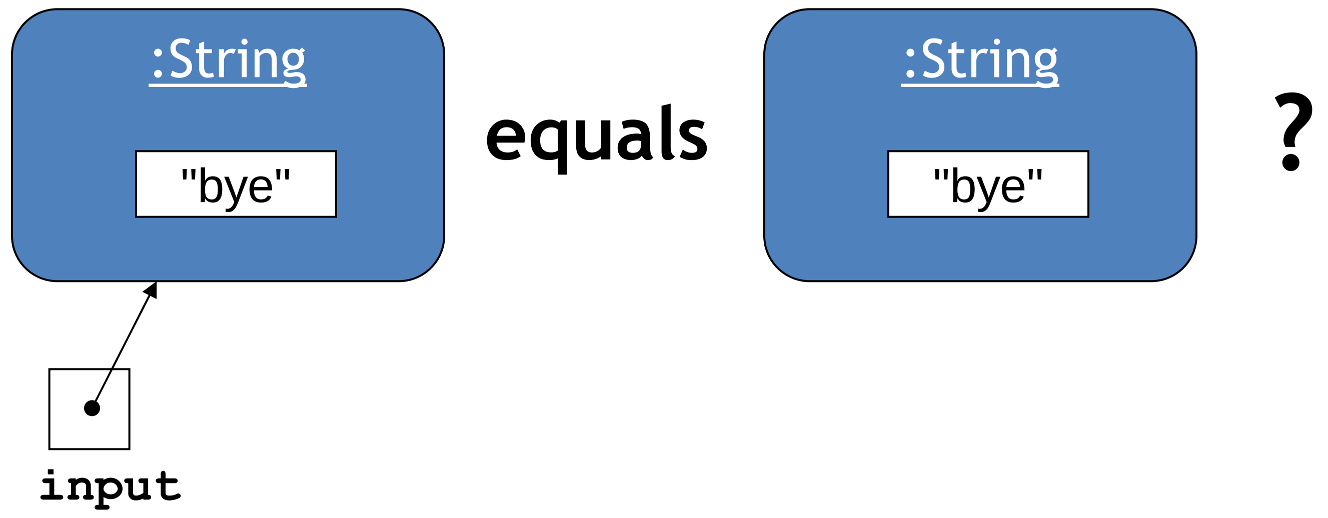


→ (may be) false!

String: Identity vs Equality (3)

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

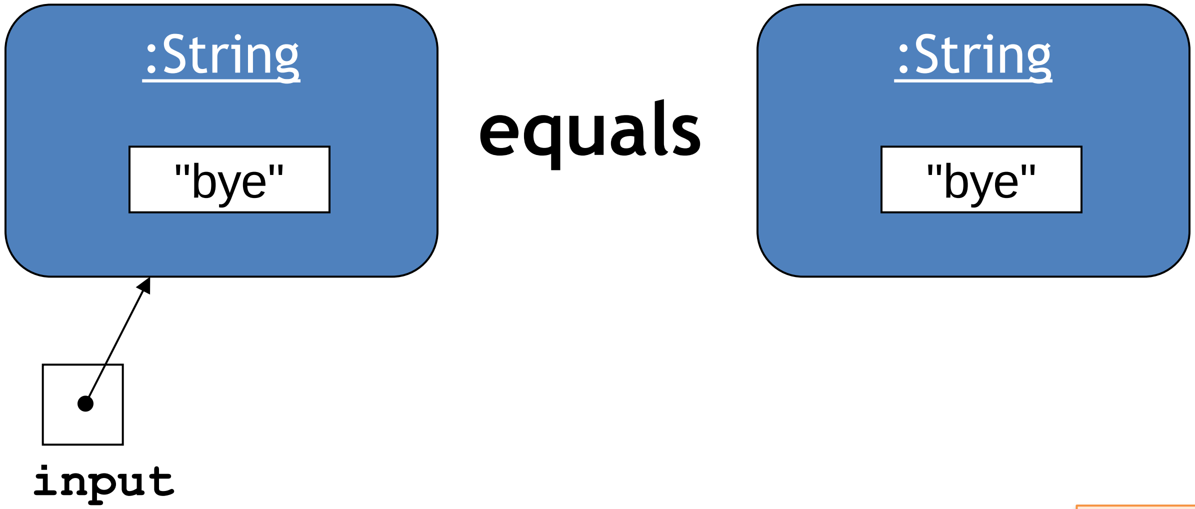
equals tests equality



String: Identity vs Equality (3)

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

equals tests equality



→ true!

Some common errors when comparing Strings...

What’s wrong here?

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

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

Strings need to use the .equals method

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

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

What’s wrong here?

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

  if(num1 = num2)
  {
    System.out.println(num1+" is the same as "+ num2);
  }
  else
  {
    System.out.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)
  {
    System.out.println(num1+" is the same as "+ num2);
  }
  else
  {
    System.out.println(num1+" is NOT same as "+ num2);
  }

}
```

Questions?

References

- Reas, C. & Fry, B. (2014) Processing – A Programming Handbook for Visual Designers and Artists, 2nd Edition, MIT Press, London.