Strings

Strings and their methods

Produced by: Dr. Siobhán Drohan
Ms Mairead Meagher
Ms Siobhan Roche

SETU Ollscoil Teicneolaíochta an Oirdheiscirt South East Technological University

Department of Computing and Mathematics
setu.ie

Topics list - **Strings**

1. Primitive Types: **char**
2. Object Types: **String**
3. Strings and **Java API**
4. Strings – **methods**
5. **Method calls**
   - **Internal**
   - **External**
   - **Dot notation**
6. Using String methods: some **examples**

Recap: 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.

- It’s values range:
  - from '\u0000‘ (or 0)
  - to '\uffff' (or 65,535)

- For example:
  - ‘A’ is ‘\u0041’
  - ‘a’ is ‘\u0061’

https://unicode-table.com/en/#control-character

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

Example 3.18 – Alphabet

```
public static void main(String[] args)
{
    char letter = 'A';
    for (int i = 1; i<=26; i++){
      System.out.println(letter);
      letter++;
}
```

```
StringExamples.main({ });
A
B
C
D
E
F
G
H
I
J
K
L
M
N
O
P
Q
R
S
T
U
V
W
X
Y
Z
```

This code uses the underlying **Unicode** value for 'A' (i.e. '\u0041')
and adds one to it each time the for loop is iterated.

As the for loop is iterated 26 times,
and the starting value is 'A',
our loop will print the alphabet to the console.

Topics list - **Strings**

1. Primitive Types: **char**
2. Object Types: **String**
3. Strings and **Java API**
4. Strings – **methods**
5. **Method calls**
   - **Internal**
   - **External**
   - **Dot notation**
6. Using String methods: some **examples**

Object types e.g. String

- Strings, which are widely used in Java programming, are a sequence of characters enclosed by double quotes (“”).
  e.g. **“seq of chars”**

- 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.
```

Object Data Types start with a Capital Letter to distinguish them from Primitive data types

Source: Reas & Fry (2014)

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.

Topics list - **Strings**

1. Primitive Types: **char**
2. Object Types: **String**
3. Strings and **Java API**
4. Strings – **methods**
5. **Method calls**
   - **Internal**
   - **External**
   - **Dot notation**
6. Using String methods: some **examples**

Strings and **Java's API**

- This link is to Java's **A**pplication **P**rogramming **I**nterface (**API**), version 8.
  https://docs.oracle.com/javase/8/docs/api/index.html?overview-summary.html

- More information on the **String**'s **methods**:
  https://docs.oracle.com/javase/8/docs/api/java/lang/String.html

Topics list - **Strings**

1. Primitive Types: **char**
2. Object Types: **String**
3. Strings and **Java API**
4. Strings – **methods**
5. **Method calls**
   - **Internal**
   - **External**
   - **Dot notation**
6. Using String methods: some **examples**

Strings - some API **methods**

Return Type	**Method** Name	Description
int	**length**()	Returns the length of this string.
String	**toLowerCase**()	Converts all of the characters in this String to lower case.
String	**toUpperCase**()	Converts all of the characters in this String to upper case.
String	**trim**()	Returns a string whose value is this string, with any *leading and trailing* whitespace removed.
String	**substring**(int beginIndex, int endIndex)	Returns a string that is a substring of this string.
char	**charAt**(int index)	Returns the char value at the specified index.

Check for others here > https://docs.oracle.com/javase/8/docs/api/java/lang/String.html

Topics list - **Strings**

1. Strings and **Java API**
2. Strings – **methods**
3. **Method calls**
   - **Internal**
   - **External**
   - **Dot notation**
4. Using String methods: some **examples**

External method calls

- We want to check the **length** of this String:

  String name = “Joe Soap”;

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

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

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

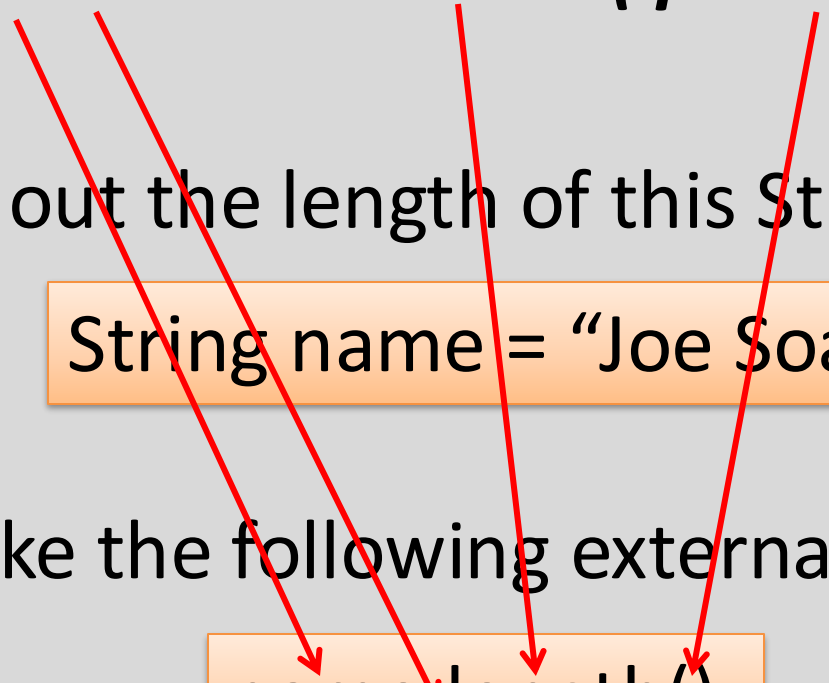
- A **call to a method of another object** is called an ***external*** method call.

  (objects {e.g.String} are usually defined in their own separate files)

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();
```

Topics list - **Strings**

1. Primitive Types: **char**
2. Object Types: **String**
3. **Primitive** Types **versus Object** Types
4. Strings and **Java API**
5. Strings - **methods**
6. **Method calls**
   - **Internal**
   - **External**
   - **Dot notation**
7. Using String methods: some **examples**

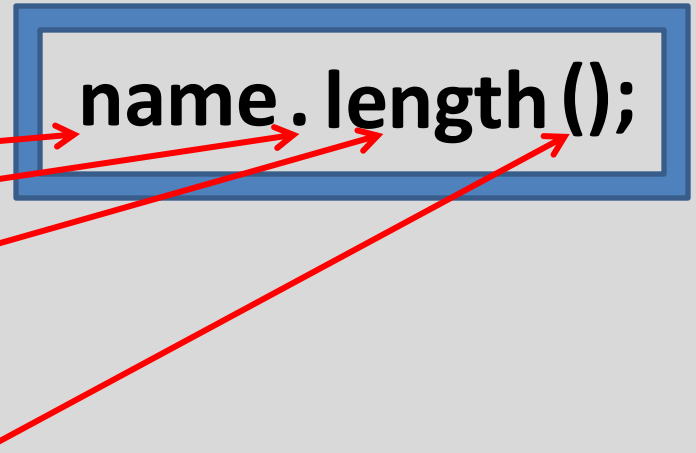
Dot Notation

- Java code can call methods of other objects using dot notation.

- The syntax is:

  object.methodname (parameter-list)

- It consists of:
  - An object
  - A dot
  - A method name
  - The parameter(s) for the method



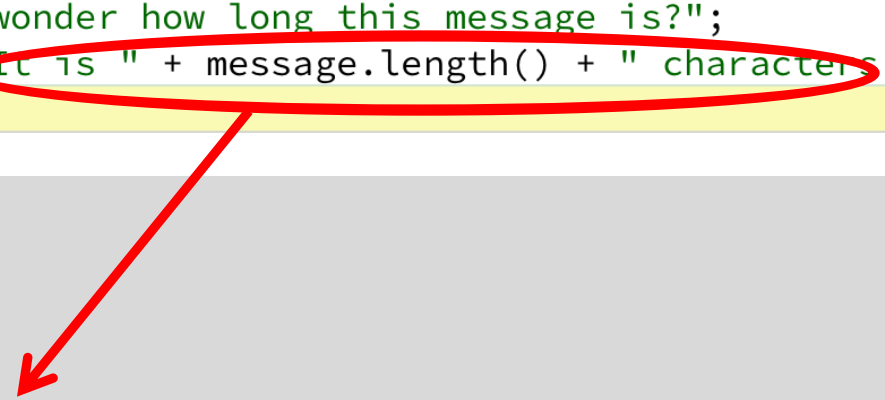
Topics list - **Strings**

1. Primitive Types: **char**
2. Object Types: **String**
3. **Primitive** Types **versus Object** Types
4. Strings and **Java API**
5. Strings - **methods**
6. **Method calls**
   - **Internal**
   - **External**
   - **Dot notation**
7. Using String methods: some **examples**

Example 1

Printing the length of a String to the console.

```
public static void main(String[] args)
{
    String message = "I wonder how long this message is?";
    System.out.println("It is " + message.length() + " characters long");
}
```

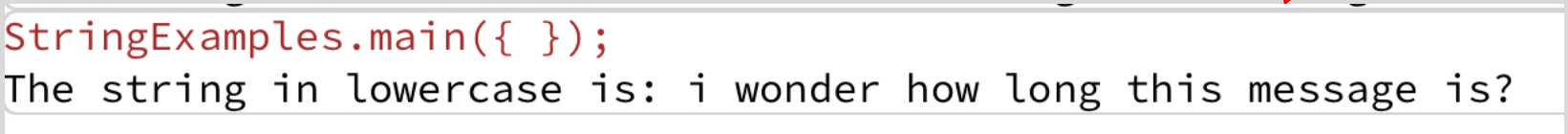


```
StringExamples.main({ });
It is 34 characters long
```

Example 2

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

```
public static void main(String[] args) {
    String message = "I wonder how long this message is?";
    System.out.println("The string in lowercase is: " + message.toLowerCase() );
}
```



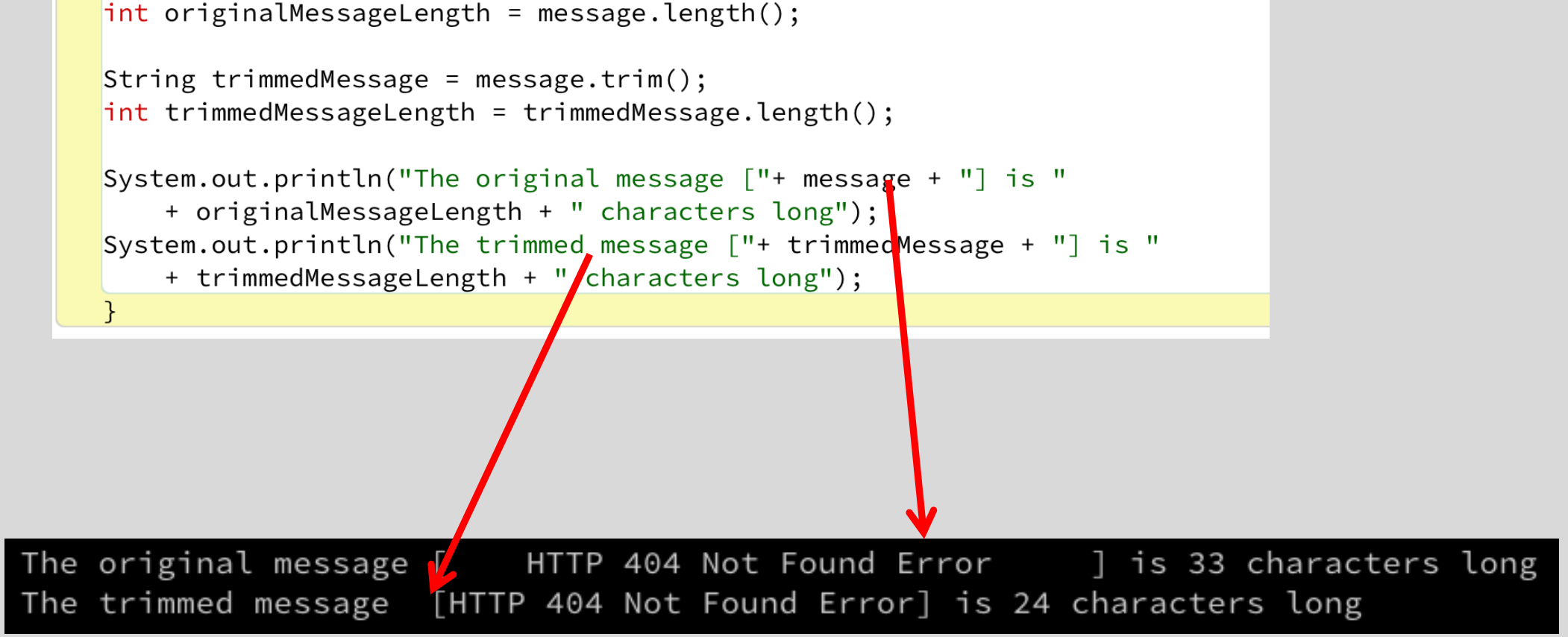
Example 3

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

```
public static void main(String[] args) {
    String message = "      HTTP 404 Not Fount Error    ";
    int originalMessageLength = message.length();

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

    System.out.println("The original message ["+ message + "] is "
        + originalMessageLength + " characters long");
    System.out.println("The trimmed message ["+ trimmedMessage + "] is "
        + trimmedMessageLength + " characters long");
    }
```



Questions?

References

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