Chapter 13

How to work with strings

Objectives

Applied

1. Use the methods of the String class to work with immutable strings.
2. Use the StringBuilder class to create and work with mutable strings.
3. Given Java code that uses any of the language elements presented in this chapter, explain what each statement does.

Knowledge

1. Explain the difference between a mutable and an immutable string and why it's usually more efficient to use a mutable string.
2. Explain how Java determines the initial capacity of a StringBuilder object and the new capacity of a StringBuilder object when its current capacity is exceeded.

The String class

```
java.lang.String;
```

How to declare and initialize string variables

```
// empty string
String productCode = "";

// string literal
String title = "Murach's Java Programming";

// same object as another variable
String bookTitle = title;
```

How to join strings

```
String name = "Bob";                    // name is "Bob"
String message = "Hi, " + name;         // "Hi, Bob"
```

How to join a string and a number

```
int years = 3;
String message = "Years: " + years;     // "Years: 3"
```

How to append one string to another

```
String name = "Bob";              // name is "Bob"
name = name + " ";                // name is "Bob "
name = name + "Smith";            // name is "Bob Smith"
```

Another way to append one string to another

```
String name = "Bob";              // name is "Bob"
name += " ";                      // name is "Bob "
name += "Smith";                  // name is "Bob Smith"
```

Methods for comparing strings

```
equals(String)
equalsIgnoreCase(String)
isEmpty()
startsWith(String)
endsWith(String)
```

A common mistake when testing for equality

```
if (productCode == "java") {
    System.out.println(
        "This does not test for equality.");
}
```

Use the equals() method to test for equality

```
if (productCode.equals("java")) {
    System.out.println("This tests for equality.");
}
```

How to use the equals() method to check for an empty string

```
if (productCode.equals("")) {
    System.out.println("You must enter a product code.");
}
```

How to use the isEmpty() method (Java 6 and later)

```
if (productCode.isEmpty()) {
    System.out.println("You must enter a product code.");
}
```

How to use the startsWith() method

```
if (productDescription.startsWith("Murach")) {
    System.out.println("This book is a Murach book.");
}
```

How to use the endsWith() method

```
if (productDescription.endsWith("Programming")) {
    System.out.println(
        "This book is about programming.");
}
```

Methods for working with string indexes

```
length()
indexOf(String)
indexOf(String, startIndex)
lastIndexOf(String)
lastIndexOf(String, startIndex)
charAt(index)
```

How to get the length of a string

```
String productCode = "java";
int length = productCode.length();    // length is 4
```

How to use the length() method to check for an empty string

```
if (productCode.length() == 0) { ... }
```

Get the index values for the two spaces

```
String name = "Martin Van Buren";
int index1 = name.indexOf(" ");           // index1 is 6
int index2 = name.indexOf(" ", index1+1); // index2 is 10
```

Another way to get the index values

```
String name = "Martin Van Buren";
int index1 = name.lastIndexOf(" ");           // 10
int index2 = name.lastIndexOf(" ", index1-1); // 6
```

Get the index of a string

```
String name = "Martin Van Buren";
int index = name.indexOf("Van");             // 7
```

Get the character at the specified index

```
String name = "Martin Van Buren";
char char1 = name.charAt(0);  // char1 is 'M'
char char2 = name.charAt(1);  // char2 is 'a'
char char3 = name.charAt(2);  // char3 is 'r'
```

Use an index in a for loop

```
String itemNumber = "RT-123";
for (int i = 0; i < itemNumber.length(); i++) { ... }
```

Methods for modifying strings

```
trim()
substring(startIndex)
substring(startIndex, endIndex)
replace(oldChar, newChar)
split(delimiter)
```

Code that trims spaces from a string

```
String choice = "  y  ";
choice = choice.trim();    // choice is "y"
```

Code that parses a string

```
String name = "Mike Murach";
int index = name.indexOf(" ");               // 4
String firstName = name.substring(0, index); // "Mike"
String lastName = name.substring(index + 1); // "Murach"
```

Code that adds dashes to a credit card number

```
String ccNumber = "4012888888881881";
String part1 = ccNumber.substring(0,4);
String part2 = ccNumber.substring(4,8);
String part3 = ccNumber.substring(8,12);
String part4 = ccNumber.substring(12,16);
ccNumber = part1 + "-" + part2 + "-" +
    part3 + "-" + part4;
```

Code that changes the separator character in a phone number

```
String phoneNumber = "977-555-1212";
phoneNumber = phoneNumber.replace("-", ".");
```

Code that removes dashes from a credit card number

```
String ccNumber = "4012-8888-8888-1881";
ccNumber = ccNumber.replace("-", "");
```

Code that stores the parts of a name in an array

```
String name = "Michael R Murach";
String[] nameParts = name.split(" ");
String firstName = nameParts[0];        // "Michael"
String middleInitial = nameParts[1];    // "R"
String lastName = nameParts[2];         // "Murach"
```

The StringBuilder class

```
java.lang.StringBuilder;
```

Constructors of the class

```
StringBuilder()
StringBuilder(capacity)
StringBuilder(String)
```

Some methods of the class

```
append(data)
capacity()
length()
```

Code that creates a credit card number

```
StringBuilder ccNumber = new StringBuilder();
ccNumber.append("4012");
ccNumber.append("8888");
ccNumber.append("8888");
ccNumber.append("1881");
```

How capacity automatically increases

```
StringBuilder name =
    new StringBuilder(8);            // capacity is 8
name.append("Raymond R. Thomas");
int length = name.length();          // length is 17
int capacity2 = name.capacity();     // capacity2 is 18
                                     // (2 * capacity1 + 2)
```

More methods of the StringBuilder class

```
insert(index, data)
replace(startIndex, endIndex, String)
delete(startIndex, endIndex)
deleteCharAt(index)
setCharAt(index, character)
charAt(index)
substring(index)
substring(startIndex, endIndex)
toString()
```

Code that adds dashes to a credit card number

```
ccNumber.insert(4, "-");
ccNumber.insert(9, "-");
ccNumber.insert(14, "-");
```

Code that removes dashes from a credit card number

```
for(int i = 0; i < ccNumber.length(); i++) {
    if (ccNumber.charAt(i) == '-') {
        ccNumber.deleteCharAt(i);
        i--;
    }
}
```

Code that parses a credit card number

```
String part1 = ccNumber.substring(0,4);
String part2 = ccNumber.substring(4,8);
String part3 = ccNumber.substring(8, 12);
String part4 = ccNumber.substring(12);
```

The console for the Product Lister application

```
Welcome to the Product Lister

Enter product code: java
Another product? (y/n): y

Enter product code: mysql
Another product? (y/n): n

Code      Description                       Price
========= ================================  =========
java      Murach's Java Programming         $57.50
mysql     Murach's MySQL                    $54.50
```

The StringUtil class

```
package murach.ui;

public class StringUtil {

    public static String pad(String s, int length) {
        if (s.length() < length) {
            // append spaces until the string is length
            StringBuilder sb = new StringBuilder(s);
            while (sb.length() < length) {
                sb.append(" ");
            }
            return sb.toString();
        } else {
            // truncate the string to the specified length
            return s.substring(0, length);
        }
    }
}
```

The ProductListerApp class

```
package murach.ui;

import murach.db.ProductDB;
import murach.business.Product;

public class ProductListerApp {

    public static void main(String args[]) {
        System.out.println(
            "Welcome to the Product Lister\n");

        final int CODE_WIDTH = 10;
        final int DESC_WIDTH = 34;
        final int PRICE_WIDTH = 10;
```

The ProductListerApp class (cont.)

```
// set up display string
StringBuilder list = new StringBuilder();
list.append(StringUtil.pad("Code", CODE_WIDTH));
list.append(StringUtil.pad("Description",
    DESC_WIDTH));
list.append(StringUtil.pad("Price", PRICE_WIDTH));
list.append("\n");

list.append(
    StringUtil.pad("=========", CODE_WIDTH));
list.append(
    StringUtil.pad(
        "=================================",
        DESC_WIDTH));
list.append(
    StringUtil.pad("=========", PRICE_WIDTH));
list.append("\n");
```

The ProductListerApp class (cont.)

```
        // perform 1 or more calculations
        String choice = "y";
        while (choice.equalsIgnoreCase("y")) {
            // get the input from the user
            String productCode =
                Console.getString("Enter product code: ");

            Product product =
                ProductDB.getProduct(productCode);

            list.append(
                StringUtil.pad(product.getCode(),
                CODE_WIDTH));
            list.append(
                StringUtil.pad(product.getDescription(),
                DESC_WIDTH));
            list.append(
                StringUtil.pad(product.getPriceFormatted(),
                PRICE_WIDTH));
            list.append("\n");
```

The ProductListerApp class (cont.)

```
                // see if the user wants to continue
                choice = Console.getString(
                    "Another product? (y/n): ");
                System.out.println();
            }
            System.out.println(list);
        }
}
```