Java Reference - Framework

By John Bentley

Table of Contents

Basic I/O – Streams

Overview

Introduction

A stream is a sequence of data. There are input and output streams.

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/essential/io/streams.html

Streams, essential tips:

- For character output favour PrintWriter as it supports: buffering; line operations; and formatting.
- For character input favour Scanner as it supports: buffering; tokenizing; conversion to native datatypes; and BigDecimal support.
- For serializing objects favour ObjectStreams. ObjectStreams allow you to serialize complex objects (as well as the primitive datatypes).
- You generally want to avoid raw (byte, character, data, or object) I/O solutions. Instead you want to consume and produce data in a structured interoperable format, for example using XML with XML classes [*fix: be specific about which XML classes].

Stream Types Syntax Summary

Byte Streams	File	import java.io.FileInputStream;
		import java.io.FileOutputStream;
		import java.io.IOException;
		
		byteStreams() throws IOException {
		
		inputStream = new FileInputStream("Xanadu.txt");
		outputStream = new FileOutputStream("XanaduOutAgain.txt");

	Buffering	[look it up yourself if you are desperate]
	System ("Standard")	System.out... System.err... System.in

Object (Byte) Streams	(buffering, wrap file stream, serialize objects)	outputStream = new ObjectOutputStream(
		new BufferedOutputStream(new FileOutputStream(dataFileName)));
		outputStream.writeObject(Calendar.getInstance());
		inputStream = new ObjectInputStream(
		new BufferedInputStream(new FileInputStream(dataFileName)));
		dateTime = (Calendar) inputStream.readObject();

Character Streams	File	// File character wrapper
		import java.io.FileReader;
		import java.io.FileWriter;
		import java.io.IOException;
		
		throws IOException, FileNotFoundException {
		
		inputStream = new FileReader("Xanadu.txt");
		outputStream = new FileWriter("XanaduOutCharacter.txt");

	Console	Console console = System.console();
		login = console.readLine("Enter your login: ");
		password = console.readPassword("Enter your password: ");
		console.printf("Login: %s%nOldPassword: %s", login, new String(password));

	Newtork Sockets [*fix ???]	InputStreamReader OutputStreamReader
Character Stream Wrappers	Buffering	inputStream = new BufferedReader(new FileReader("Xanadu.txt")); outputStream = new BufferedWriter(new FileWriter("XanaduOut.txt"));
	Scanning	scanner = new Scanner(new BufferedReader(new FileReader(inputFileName)));
	PrintWriter (EOL; buffering; and formatting)	import java.io.PrintWriter; outputStream2 = new PrintWriter(new FileWriter("XanaduOutCharacterBufferedLineOrientated.txt"));

General Stream Coding

Always close streams.

```
try {

  inputStream = new FileInputStream("Xanadu.txt");
  outputStream = new FileOutputStream("XanaduOutAgain.txt");

  // ...

} finally {
  if (inputStream != null) {
    inputStream.close();
  }
  if (outputStream != null) {
    outputStream.close();
  }
}
```

Closing streams frees up resources.

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/essential/io/bytestreams.html

Stream Types

Introduction

A stream supports different kinds of data: bytes, localized characters, primitive datatypes and objects.

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/essential/io/bytestreams.html

Byte and Character Streams

Byte streams operate at a low level (dealing with bytes) and you should generally avoid using it.

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/essential/io/bytestreams.html

Character streams automatically stores and retrieves data according to the local character set. For Western locales, the character set is "usually an 8 bit superset of ASCII".

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/essential/io/charstreams.html

Character streams are usually wrappers for byte streams

```
// File character wrapper
inputStream = new FileReader("Xanadu.txt");
outputStream = new FileWriter("XanaduOutCharacter.txt");

// Network Sockets Wrapper (*fix)
InputStreamReader
OutputStreamReader
```

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/essential/io/charstreams.html

Data (type) Streams

Data streams support binary I/O of primitive datatypes (`boolean`, `char`, `byte`, `short`, `int`, `long`, `float`, and `double`) as well as String values.

Object Streams

Use object streams over data streams.

Object Streams inherit from Data Streams, so have the power of Data Streams, and have the ability to serialize objects.

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/essential/io/datastreams.html &
(Oracle, 2017) http://docs.oracle.com/javase/tutorial/essential/io/objectstreams.html

Object streams can serialize most objects.

In order to serialize an object the object's class must have implemented the Serializable interface.

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/essential/io/objectstreams.html and
(Oracle, 2011) http://docs.oracle.com/javase/6/docs/api/java/io/Serializable.html

Object Stream Example

```
import java.io.BufferedInputStream;
import java.io.BufferedOutputStream;
import java.io.EOFException;
import java.io.FileInputStream;
import java.io.FileNotFoundException;
import java.io.FileOutputStream;
import java.io.IOException;
import java.io.ObjectOutputStream;
import java.io.ObjectInputStream;
import java.math.BigDecimal;
import java.util.Calendar;

public class ObjectSreamsDemo {

  // pty for "proprietary"
  static final String dataFileName = "ObjectData.pty";
```

```
  static final BigDecimal[] prices = { new BigDecimal("19.99"), new BigDecimal("9.99"),
      new BigDecimal("15.99"), new BigDecimal("3.99"), new BigDecimal("4.99") };
  static final int[] units = { 12, 8, 13, 29, 50 };
  static final String[] description = { "Java T-shirt", "Java Mug", "Duke Juggling Dolls",
      "Java Pin", "Java Key Chain" };

  public static void start() throws FileNotFoundException, IOException,
ClassNotFoundException {

    ObjectOutputStream outputStream = null;
    try {
      outputStream = new ObjectOutputStream(
          new BufferedOutputStream(new FileOutputStream(dataFileName)));
      outputStream.writeObject(Calendar.getInstance());
      for (int i = 0; i < prices.length; i++){
        outputStream.writeObject(prices[i]);
        outputStream.writeInt(units[i]);
        outputStream.writeUTF(description[i]);
      }
    } finally {
      if (outputStream != null) {
        outputStream.close();
      }
    }

    ObjectInputStream inputStream = null;
    try {
      inputStream = new ObjectInputStream(
                      new BufferedInputStream(new FileInputStream(dataFileName)));
      Calendar dateTime = null;
      BigDecimal price;
      int unit;
      String description;
      BigDecimal total = new BigDecimal(0);

      dateTime = (Calendar) inputStream.readObject();
      System.out.printf("%1$tF %1$tR%n", dateTime);

      try {
        while (true) {
          price = (BigDecimal) inputStream.readObject();
          unit = inputStream.readInt();
          description = inputStream.readUTF();
          System.out.printf("%02d units of %s at $%.2f%n", unit, description, price);
          total = total.add(price.multiply(new BigDecimal(unit)));
        }
      } catch (EOFException exn) {
        System.out.format("For a TOTAL of: $%.2f%n", total);
      }

    } finally {
      if (inputStream != null) {
        inputStream.close();
      }
    }
  } // start()
} // ObjectStreamDemo

// output
2013-07-29 05:39
12 units of Java T-shirt at $19.99
08 units of Java Mug at $9.99
13 units of Duke Juggling Dolls at $15.99
29 units of Java Pin at $3.99
50 units of Java Key Chain at $4.99
For a TOTAL of: $892.88
```

(Bentley, TutorialAtOracle Code Examples, 2013) ObjectStreamsDemo.java
Adapted from (Oracle, 2017)
http://docs.oracle.com/javase/tutorial/displayCode.html?code=http://docs.oracle.com/javase/tutorial/essential/io/examples/ObjectStreams.java

Things to do with streams

Buffering

When using streams use buffering. There are two buffered byte streams: BufferedInputStream and BufferedOutputStream. There are three buffered character streams: BufferedReader; BufferedWriter and PrintWriter.

Buffered streams use resources (disk access, network activity, etc) more efficiently.

(Oracle, 2017) *http://docs.oracle.com/javase/tutorial/essential/io/buffers.html*

You code buffers by wrapping unbuffered streams.

```
inputStream = new BufferedReader(new FileReader("Xanadu.txt"));
outputStream = new BufferedWriter(new FileWriter("XanaduOut.txt"));
```

(Oracle, 2017) *http://docs.oracle.com/javase/tutorial/essential/io/buffers.html*

Some buffers support autoflushing

```
// Enable autoflush.
outputStream = new PrintWriter(new FileWriter("FileOut.txt"), true);

// Every println or format will now flush.
```

(Oracle, 2017) *http://docs.oracle.com/javase/tutorial/essential/io/buffers.html*

However it might make more sense, depending on the app, to flush manually. Use flush()

```
outputStream.flush();
```

(Oracle, 2017) *http://docs.oracle.com/javase/tutorial/essential/io/buffers.html*

Otherwise the buffer is flushed when it is full or when the output stream is close. It might be most efficient simply to let the buffer flush when you close the stream.

```
outputStream = new BufferedWriter(new FileWriter("FileOut.txt"));

...

if (outputStream != null) {
  outputStream.close();  // Flushes the buffer
}
```

Line Operations

When using character streams it is often a good idea to use a line orientated approach (rather than at one character at a time).

```
// Using BufferedWriter
inputStream = new BufferedReader(new FileReader("Xanadu.txt"));
outputStream =
    new BufferedWriter(new FileWriter("XanaduOutCharacterBufferedLineOrientated.txt"));

String line;
while ((line = inputStream.readLine()) != null) {
  System.out.println(line);
  outputStream.write(line);
  outputStream.newLine();
}
```

(Bentley, TutorialAtOracle Code Examples, 2013) StreamsDemo.java

PrintWriter is ideal for: buffering; line operations; and formatting. For the formatting features favour it over BufferedWriter.

```
PrintWriter outputStream = null;

// Using the PrintWriter wrapper
inputStream = new BufferedReader(new FileReader("Xanadu.txt"));
outputStream =
    new PrintWriter(new FileWriter("XanaduOutCharacterBufferedLineOrientated.txt"));

String line;
while ((line = inputStream.readLine()) != null) {
  System.out.println(line);
  outputStream.println(line);
}
```

(Bentley, TutorialAtOracle Code Examples, 2013) StreamsDemo.java

Scanning

Objects of type Scanner are useful for breaking down formatted input into tokens and translating individual tokens according to their data type.

Persons.txt

```
Joe ~ Blower ~ 30 ~ Cool ~ 12,234.24
Mary ~     Contrary ~ 24 ~ Fool ~ 123098.34
```

Code ...

```
import java.io.BufferedReader;
import java.io.FileNotFoundException;
import java.io.FileReader;
import java.io.IOException;
import java.math.BigDecimal;
import java.util.Locale;
import java.util.Scanner;

public static void scanAndDataTypeConvert() throws FileNotFoundException, IOException,
    InvalidDataException {

  Scanner scanner = null;
  String inputFileName = "Persons.txt";

  // A local class
  class Person {
    String givenName = "";
    String familyName = "";
    int age = 0;
    String description = "";
    BigDecimal wage = null;

    // Overrides Object.toString()
    public String toString() {
      return String.format("%5s %10s %d %s $%,10.2f", this.givenName, this.familyName,
          this.age, this.description, this.wage);
    }
  } // Person

  // Declare and Initialise
  Person[] persons = new Person[2];

  try {

    scanner = new Scanner(new BufferedReader(new FileReader(inputFileName)));
```

```
    scanner.useLocale(new Locale("en", "AU"));

    // Any number of spaces surrounding a tilde "~" or any number of spaces till End
    // of Line.
    scanner.useDelimiter("(\\s*~\\s*)|(\\s*(\r\n))");

    int i = 0;
    while (scanner.hasNext()) {
      persons[i] = new Person();

      persons[i].givenName = scanner.next();
      persons[i].familyName = scanner.next();

      if (scanner.hasNextInt()) {
        persons[i].age = scanner.nextInt();
      } else {
        throw new InvalidDataException("Check" + inputFileName
            + ". It contains data in an invalid sequence");
      }

      persons[i].description = scanner.next();

      if (scanner.hasNextBigDecimal()) {
        persons[i].wage = scanner.nextBigDecimal();
      } else {
        throw new InvalidDataException("Check" + inputFileName
            + ". It contains data in an invalid sequence");
      }

      i++;
    } // while

  } finally {

    if (scanner != null) {
      scanner.close();
    }

  } // Try

  for (Person person : persons) {
    System.out.println(person.toString());
  }

} // scanAndDataTypeConvert
```

Output ...

```
  Joe     Blower 30 Cool $ 12,234.24
 Mary   Contrary 24 Fool $123,098.34
```

(Oracle, 2017) *http://docs.oracle.com/javase/tutorial/essential/io/scanning.html*

Scanning Line Operations

Example Scanning line by line.

```
/**
 * Parse Text File, and operate on a line by line basis. Output to screen.
 *
 * This is incidental to XML operations, it is just demoing line by line parsing.
 *
 * @param txtFileName
 * @throws UnexpectedException
 */
static public void ParseTextFileByLineOutputToScreen(
        String txtFileName) throws UnexpectedException {
    File txtFile = new File(txtFileName);

    BufferedReader bufferedReader = null;
    Scanner scanner = null;
```

```
    try {
        bufferedReader = new BufferedReader(new FileReader(txtFile));
        scanner = new Scanner(bufferedReader);

    } catch (FileNotFoundException e) {
        e.printStackTrace();
    }

    PrintWriter printWriter = null;
    printWriter = new PrintWriter(System.out);


    try {
        String field;
        String value;
        // Find one or more characters that are not "~" and not a space.
        String regexPattern = "[^~\\s]+";
        while (scanner.hasNextLine()) {
            field = scanner.findInLine(regexPattern);
            // findInLine consumes that which it matches. So we can apply the same
            // regex pattern to consume more of the line.
            value = scanner.findInLine(regexPattern);
            scanner.nextLine();
            printWriter.printf("%s = %s%n", field, value);
        } // while
        printWriter.flush();
    } finally {
        if (scanner != null) {
            scanner.close();
        }
        if (printWriter != null) {
            printWriter.close();
        }
    } // Try
}

// Input. takeOffAndLandingWeatherReportValuePairs.txt
location~ YMML
dateTimeUtc~ 2003-05-16T16:00:00Z
wind~ 06014
direction~ 060
speed~ 14
qnh~ 1013.2
temperature~ 05
dewPoint~ 02
sky~
cavok~
remarks~ Runway 16L closed for repair.

// Output (to Screen)
txtFileName:takeOffAndLandingWeatherReportValuePairs.txt

location = YMML
dateTimeUtc = 2003-05-16T16:00:00Z
wind = 06014
direction = 060
speed = 14
qnh = 1013.2
temperature = 05
dewPoint = 02
sky = null
cavok = null
remarks = Runway
```

C:\Users\John\Documents\Sda\Code\BjaxXsl\Examples\XmlAviationWeather\JaxpDemo\src\sax\SaxMain.java

Formatting

If you want to use formatting methods, like format and printf, then use the PrintWriter. BufferedWriter.

(Bentley, TutorialAtOracle Code Examples, 2013) StreamsDemo.java &
(Oracle, 2012) http://docs.oracle.com/javase/tutorial/essential/io/formatting.html

For byte stream output there is also the PrintStream class. In practice you'd only use this indirectly by using System.out and System.err.

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/essential/io/formatting.html

For printf style formatting see

(Bentley, Java Reference - Language - Printf Style Formatting.docx, 2013)

Stream sources and destinations

Standard Streams

Stream	Notes	Code Example
System.out	PrintStream objects. A byte stream with character features.	
System.err		
System.in	Byte stream with no character features.	InputStreamReader cin = new InputStreamReader(System.in);

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/essential/io/cl.html

Generally don't bother with System.in, use the Console instead.

The Console

The console is: a character stream; and useful for password input.

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/essential/io/cl.html
See (Bentley, Java Reference - Language.docx, 2017), "Running console applications created by eclipse".

Console example.

```
public static void main(String[] args) {
  Console console = System.console();
  String login = "";
  char[] password;

  if (console == null) {
      System.err.println("No console.");
      System.exit(1);
  }

  login = console.readLine("Enter your login: ");
  password = console.readPassword("Enter your password: ");

  console.printf("Login: %s%nOldPassword: %s", login, new String(password));
}
```

(Bentley, TutorialAtOracle Code Examples - PasswordConsole, 2013)

For a console example closer to real world use see ...

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/essential/io/cl.html

Reference Example (Using TAF)

```
private static void streamsReferenceExampleTAF() throws FileNotFoundException {
  Scanner scanner = null;
  String inputFileName = "TafYssy.txt";
  String line = "";
  String utcDateTimeString = ""; // E.g. 120834

  try {
    scanner = new Scanner(new BufferedReader(new FileReader(inputFileName)));

    scanner.useLocale(new Locale("en", "AU"));

    SimpleDateFormat sdfInput = new SimpleDateFormat("ddHHmm");
    sdfInput.setTimeZone(TimeZone.getTimeZone("UTC"));

    SimpleDateFormat sdfOutput = new SimpleDateFormat("yyyy-MM ddHHmm Z z");

    Calendar nowCalendar = GregorianCalendar.getInstance();

    while (scanner.hasNextLine()) {
      line = scanner.nextLine();
      utcDateTimeString = matchedSubstring(line, "\\d{6}");

      if (utcDateTimeString.length() > 0) {
        Date localDateTime = new Date();
        localDateTime = sdfInput.parse(utcDateTimeString);
        Calendar localDateTimeCalendar = GregorianCalendar.getInstance();
        localDateTimeCalendar.setTime(localDateTime);
        localDateTimeCalendar.set(Calendar.YEAR, nowCalendar.get(Calendar.YEAR));
        localDateTimeCalendar.set(Calendar.MONTH, nowCalendar.get(Calendar.MONTH));
        localDateTime = localDateTimeCalendar.getTime();

        System.out.printf("%s : %s UTC (%s)%n", line, utcDateTimeString,
            sdfOutput.format(localDateTime));
      } else {
        System.out.println(line);
      }
    }

  } catch (ParseException e) {
    // TODO Auto-generated catch block
    e.printStackTrace();
  } finally {

    if (scanner != null) {
      scanner.close();
    }
  } // Try

} // streamsReferenceExampleTAF

// Output
TAF YSSY 011103Z 0112/0218 : 011103 UTC (2013-09 012103 +1000 EST)
03007KT CAVOK
FM011400 33007KT 9000 FU NSC : 011400 UTC (2013-09 020000 +1000 EST)
FM012100 33007KT 9999 FEW030 : 012100 UTC (2013-09 020700 +1000 EST)
FM020100 05012KT 9999 FEW040 : 020100 UTC (2013-09 021100 +1000 EST)
FM021600 19005KT 9999 FEW020 : 021600 UTC (2013-09 030200 +1000 EST)
RMK
T 17 15 14 15 Q 1028 1029 1028 1030
```

Basic I/O – Files

See also

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/essential/io/legacy.html

List files in a directory.

```
public static void listFilesInDirectory(String pathString) {
  // A local class (a class defined inside a block, here a method).
  class MyFilter implements FileFilter {
    @Override
    public boolean accept(File file) {
      return !file.isHidden() && file.getName().endsWith(".txt");
    }
  }

  File directory = new File(pathString);
  File[] files = directory.listFiles(new MyFilter());

  for (File fileLoop : files) {
    System.out.println(fileLoop.getName());
  }
}

// Call it
listFilesInDirectory("C:\\Users\\John\\Documents\\zTemp");

// Output
cool.txt
RedditKinsey.txt
```

(Bentley, TutorialAtOracle Code Examples, 2013) FileIODemo.java

Create a file.

```
private static void createFile(String pathSlashFileName) throws IOException {
  File newFile = new File(pathSlashFileName);
  if (!newFile.exists()) {
    newFile.createNewFile();
    System.out.printf("%s created.%n", pathSlashFileName);
  } else {
    System.out.printf("%s already exists.%n", pathSlashFileName);
  }
}

// Call it
createFile("C:\\Users\\John\\Documents\\zTemp\\CoolFile.txt");

// Output
C:\Users\John\Documents\zTemp\CoolFile.txt created.
```

(Bentley, TutorialAtOracle Code Examples, 2013) FileIODemo.java

Delete a file.

```
private static void deleteFile(String pathSlashFileName)  {
  File targetFile = new File(pathSlashFileName);
  if (targetFile.exists()) {
    targetFile.delete();
    System.out.printf("%s deleted.%n", pathSlashFileName);
  } else {
    System.out.printf("%s did not exist, so it wasn't deleted.%n", pathSlashFileName);
  }
}

// Call it
deleteFile("C:\\Users\\John\\Documents\\zTemp\\CoolFile.txt");
```

```
// Output
C:\Users\John\Documents\zTemp\CoolFile.txt deleted.
```

(Bentley, TutorialAtOracle Code Examples, 2013) FileIODemo.java

Create a directory.

```
private static void createDir(String pathString) throws IOException {
  File newDir = new File(pathString);
  if (!newDir.exists()) {
    newDir.mkdir();
    System.out.printf("%s created.%n", pathString);
  } else {
    System.out.printf("%s already exists.%n", pathString);
  }
}

// Call it
createDir("C:\\Users\\John\\Documents\\zTemp\\CoolDir");

// Output
C:\Users\John\Documents\zTemp\CoolDir created.
```

(Bentley, TutorialAtOracle Code Examples, 2013) FileIODemo.java

Internationalisation

Conceptual Overview

Internationalisation is designing an app to be readily adaptable to various languages and regions. **Localisation** is adapting an app for a specific language and/or region.

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/i18n/intro/index.html

Checklist

When internationalising an app you need to identify culturally dependent data:

- Messages
- Labels on GUI components
- Online help
- Sounds
- Colors ?
- Graphics (jb: e.g. flags)
- Icons
- Dates
- Times
- Numbers
- Currencies
- Measurements
- Phone numbers
- Honorifics and personal titles
- Postal addresses
- Page layouts ?

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/i18n/intro/checklist.html

Internationalising an app entails:

- Isolating translatable text into resource bundles.
- Formatting numbers and currencies.
- Formatting dates and times.
- Dealing with compound messages.
- Handling plurals.
- Testing characters with Unicode properties.
- Comparing strings properly.
- Converting Non-Unicode text.

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/i18n/intro/checklist.html

The Locale

Setting the Locale

Of the various ways to create a locale object the two most common are via the Locale constructor and with Locale constants. However, use the constructor technique as the constants are limited.

```
Locale fr_CA_Locale = new Locale("fr", "CA");
System.out.println(fr_CA_Locale);

// Output
fr_CA
```

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/i18n/locale/create.html &
(Bentley, TutorialAtOracle Code Examples, 2013) InternationalisationDemo.java

When creating a Locale object there are up to three codes passed to the Locale constructor. Only the first is mandatory:

- Language code. Two (preferred) or three lowercase letters. From ISO 639-2 (but ISO 639-1 two letter codes preferred). http://www.loc.gov/standards/iso639-2/php/code_list.php
- Region code. This generally distinguishes regional differences within one language. Two (preferred) or three UPPERCASE letters. From ISO 3166 or UN M.49. http://www.chemie.fu-berlin.de/diverse/doc/ISO_3166.html
- Variant code. This further distinguishes any variances in the region (e.g. regional dialects). http://www.iana.org/assignments/language-subtag-registry (search for "variant").

Variant example.

```
private static void localeCreationWithVariant() {
  double number = 234.56;

  Locale[] locales = { new Locale("th", "TH"), new Locale("th", "TH", "TH") };

  for (Locale locale : locales) {
    NumberFormat nf = NumberFormat.getNumberInstance(locale);
    System.out.printf("%s : %s%n", locale.toString(), nf.format(number));
  }
}

// output (eclipse console must be set to UTF-8)
th_TH : 234.56
th_TH_TH : ๒๓๔.๕๖
```

By convention, a NumberFormat object for the th and th_TH locales will use common Arabic digit shapes, or Arabic numerals, to format Thai numbers. However, a NumberFormat for the th_TH_TH locale uses Thai digit shapes.

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/i18n/locale/create.html &
(Bentley, TutorialAtOracle Code Examples, 2013) InternationalisationDemo.java

Codes for the most popular languages

Language	**Language code (**ISO 639-1)	**% of world population**
Mandarin	**zh (Chinese)**	**14.1%**
Spanish	**es**	**5.85%**
English	**en**	**5.52%**
Hindi	hi	4.46%

Language	Language code (ISO 639-1)	% of world population
Arabic	**ar**	**4.23%**
Portuguese	**pt**	**3.08%**
Bengali	bn	3.05%
Russian	**ru**	**2.42%**
Japanese	**ja**	**1.92%**
Punjabi	pa	1.44%
German	**de**	**1.39%**
Javanese	jv	1.25%
Wu	wu	1.20%
Malay/Indonesian	ml	1.16%
Telugu	te	1.15%
Korean	**ko**	**1.14%**
Vietnamese	vi	1.14%
French	**fr**	**1.12%**

http://en.wikipedia.org/wiki/List_of_languages_by_number_of_native_speakers#Nationalencyklopedin_.282007.29 & http://www.loc.gov/standards/iso639-2/php/code_list.php

Codes for selected language and region combos

Language (Region)	Code (language_REGION)
Spanish (Spain)	es_ES
English (United States)	en_US
English (United Kingdom)	en_GB
English (Australia)	en_AU
French (France)	fr_FR
French (Canada)	fr_CA
German (Germany)	de_DE

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/i18n/locale/create.html &
(Bentley, TutorialAtOracle Code Examples, 2013) InternationalisationDemo.java

The Locale object can be created with any valid code combination but classes that consume a Locale object generally only support a limited number of code combinations. Test what Locales are supported, for a particular worker class, with getAvailableLocales()

```
int i = 1;
for (Locale locale : DateFormat.getAvailableLocales()) {
  System.out.printf("%03d %8s : %s%n", i++, locale.toString(), locale.getDisplayName());
}

// Output
001    ja_JP : Japanese (Japan)
002    es_PE : Spanish (Peru)
003       en : English
...
```

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/i18n/locale/identify.html &
(Bentley, TutorialAtOracle Code Examples, 2013) InternationalisationDemo.java

Locale also has a getAvailalbeLocales()

```
for (Locale locale : Locale.getAvailableLocales())
```

(Bentley, TutorialAtOracle Code Examples, 2013) InternationalisationDemo.java

Default Locale

Generally use the default Locale.

The default locale is set by the (localised) host environment. Only in special cases would you need to set different Locales when creating an app. For example, when creating an app that handles multiple languages/regions simultaneously (e.g. a world time; or translation app).

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/i18n/locale/scope.html

You should never set the default Locale programmatically.

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/i18n/locale/scope.html

Get the default Locale.

```
Locale defaultLocale = Locale.getDefault();
System.out.printf("The default Locale. %s : %s%n", defaultLocale.toString(),
    defaultLocale.getDisplayName());

// Output
The default Locale. en_AU : English (Australia)
```

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/i18n/locale/scope.html &
(Bentley, TutorialAtOracle Code Examples, 2013) InternationalisationDemo.java

Be aware of client V server design considerations when handling locale-sensitive data.

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/i18n/locale/scope.html

Custom Locales

You can create custom locales, with concomitant worker classes (E.g. DateFormat, BreakIterator, Currency symbols, Time Zone names, etc), by implementing "Locale Sensitive Service Provider Interfaces (SPIs)".

```
// For example, if you would like to provide a NumberFormat object for a new locale, you
have to implement the java.text.spi.NumberFormatProvider class

// You need to extend this class and implement its methods:

    getCurrencyInstance(Locale locale)
    getIntegerInstance(Locale locale)
    getNumberInstance(Locale locale)
    getPercentInstance(Locale locale)

Locale loc = new Locale("da", "DK");
NumberFormat nf = NumberFormatProvider.getNumberInstance(loc);
```

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/i18n/locale/services.html

Isolating Locale specific data

RescourceBundle Overview

Isolate locale specific data with ResourceBundles.

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/i18n/resbundle/index.html

ResourceBundles are a set of text files or java classes. You create a RescourceBundle for each locale. You also have a default, base, RescourceBundle. Conceptual example:

```
ButtonLabel
ButtonLabel_de
ButtonLabel_en_GB
ButtonLabel_fr_CA_UNIX
```

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/i18n/resbundle/concept.html

To select the right RescouceBundle use ResourceBundle.getBundle()

```
Locale currentLocale = new Locale("fr", "CA", "UNIX");
ResourceBundle introLabels = ResourceBundle.getBundle("ButtonLabel", currentLocale);
```

ResourceBundle.getBundle() performs a search for the closest match, from more specific to more general. The search is performed first with any Locale passed to getBundle() and then with the default Locale.

```
// If en US is the default Locale and the specified Locale passed to getBundle() is
ButtonLabel_fr_CA_UNIX (as above), the search order is:

ButtonLabel_fr_CA_UNIX
ButtonLabel_fr_CA
ButtonLabel_fr
ButtonLabel_en_US
ButtonLabel_en
ButtonLabel

// Note that getBundle looks for classes based on the default Locale before it selects the
base class (ButtonLabel)
```

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/i18n/resbundle/concept.html

Always provide a default RescourceBundle with no suffixes.

Otherwise if getBundle fails to find a match a MissingResourceException will be thrown.

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/i18n/resbundle/concept.html

There are PropertyResourceBundle and ListResourceBundle subclasses of ResourceBundle. PropertyResourceBundle is backed by text files. `ListResourceBundle` is backed by java class files. PropertyResourceBundle is for use with strings only. If you need to store another type of object use a `ListResourceBundle`.

PropertyResourceBundle

PropertyResourceBundle example.

```
// In helloworld/resources/LogoLabel.properties
// For this example we won't have a LogoLabel_en_GB.properties
OkKey = OK_properties_base
CancelKey = Cancel_properties_base

// Elsewhere (perhaps in a different package).
import java.util.ResourceBundle;
...

// Normally you don't specify a Locale, but just use the default Locale.
Locale specifiedLocale = new Locale("en", "GB");
// Specify package and resource bundle base name.
ResourceBundle logoLabel =
    ResourceBundle.getBundle("helloworld.resources.LogoLabel", specifiedLocale);
System.out.println(logoLabel.getString("OkKey"));

// Output
OK_properties_base
```

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/i18n/resbundle/concept.html
(Bentley, TutorialAtOracle Code Examples, 2013) InternationalisationDemo.java

Property files are key "=" value pairs. Comments start with a hash "#".

```
# This is the LabelsBundle_de.properties file
s1 = Computer
s2 = Platte
s3 = Monitor
```

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/i18n/resbundle/propfile.html

ListResourceBundle

ListResourceBundle example. Be sure to use the 'public' access modifier for the resource List class and reference the base name fully qualified with the package name.

```
// In helloworld/resources/ButtonLabel_en_GB
package helloworld.recources;

import java.util.*;

// Don't omit the 'public' access modifier
public class ButtonLabel_en_GB extends ListResourceBundle {
  // English version
  public Object[][] getContents() {
      return contents;
  }
  static final Object[][] contents = {
      {"OkKey", "OK_en_GB"},
      {"CancelKey", "Cancel_en_GB"},
  };
}

// Elsewhere (perhaps in a different package).
import java.util.ResourceBundle;
...

// Normally you don't specify a Locale, but just use the default Locale.
Locale specifiedLocale = new Locale("en", "GB");
// Specify package and resource bundle base name.
ResourceBundle buttonLabel =
    ResourceBundle.getBundle("helloworld.resources.ButtonLabel", specifiedLocale);
System.out.println(buttonLabel.getString("OkKey"));

// Output
OK_en_GB
```

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/i18n/resbundle/concept.html
(Bentley, TutorialAtOracle Code Examples, 2013) InternationalisationDemo.java

Storing and retrieving objects with ListResourceBundle. Use getObject() for retrieval and cast the returned object to the specific type.

```
// helloworld.resources.Statistics_ja_JP.java
package helloworld.resources;

import java.util.*;

public class Statistics_ja_JP extends ListResourceBundle {
  public Object[][] getContents() {
    return contents;
  }

  private Object[][] contents = {
      { "GDP", new Integer(21300) },
      { "Population", new Integer(125449703) },
      { "Literacy", new Double(0.99) },
  };
}

// helloworld.resources.Statistics_en_CA.java is analogous
```

```
Locale[] supportedLocales = {
    new Locale("en", "CA"),
    new Locale("ja","JP")
};

for (Locale currentLocale : supportedLocales) {
  ResourceBundle statistics = ResourceBundle.getBundle("helloworld.resources.Statistics",
currentLocale);
  System.out.println(currentLocale.toString());
  System.out.printf("GDP: %d%n", (Integer) statistics.getObject("GDP"));
  System.out.printf("Population: %d%n", (Integer) statistics.getObject("Population"));
  System.out.printf("Literacy: %f%n", (Double) statistics.getObject("Literacy"));
  System.out.println();
}

// Output
en_CA
GDP: 24400
Population: 28802671
Literacy: 0.970000

ja_JP
GDP: 21300
Population: 125449703
Literacy: 0.990000
```

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/i18n/resbundle/list.html
(Bentley, TutorialAtOracle Code Examples, 2013) InternationalisationDemo.java

American English Versus The Queen's English (en_GB)

Program keywords and properties in American English.

```
// "Color" not "Colour"
```

Most computer languages will require you to American English. You'll have no choice.

Make the base RescourceBundle in the Queen's English (en_GB) and regional English variations, like American English (en_US), a specific ResourceBundle. You only need to but regional differences in the specific ResourceBundle, not repeat the whole base file.

```
// MyLabels_en_US.properties
ColorLabel = Color
```

```
// MyLabels.properties
ColorLabel = Colour
TelephoneLabel = Telephone

...
Locale specifiedLocale = new Locale("en", "US");
ResourceBundle labelRB =
    ResourceBundle.getBundle("helloworld.resources.MyLabels", specifiedLocale);
System.out.println(labelRB.getString("ColorLabel"));
System.out.println(labelRB.getString("TelephoneLabel"));

// Output. Note the specifiedLocale is "en_US" but "Telephone" comes from the base
ResourceBundle.
Color
Telephone
```

(Bentley, TutorialAtOracle Code Examples, 2013) InternationalisationDemo.java

Formatting Numbers, Currencies and Percentages

For locale sensitive formatting of numbers, currencies and percentages you have two broad options:

- Predefined formats: with Printf style formatting; or NumberFormat (described below);
- Custom formatting: with Printf style formatting; or DecimalFormat (described below), and DecimalFormatSymbols (described below).

(Bentley, Java Reference - Language - Printf Style Formatting.docx, 2013)

Predefined Format

Numbers

Format int and double numbers with NumberFormat.

```
private static void demoNumberFormat() {
  Locale[] locales =
      { new Locale("fr", "FR"), new Locale("de", "DE"), new Locale("en", "GB") };

  Integer population = new Integer(12345);
  Double height = new Double(345789.2392345);
  int fleetQuantity = 24987;
  double distance = 234567.893483;

  NumberFormat numberFormat = null;
  for (Locale localeLoop : locales) {
    numberFormat = NumberFormat.getNumberInstance(localeLoop);
    System.out.printf("%s: %s | %s | %s | %s%n",
        localeLoop.toString(),
        numberFormat.format(population),
        numberFormat.format(height),
        numberFormat.format(fleetQuantity),
        numberFormat.format(distance)
    );
  }
}

// Output
fr_FR: 12 345 | 345 789,239 | 24 987 | 234 567,893
de_DE: 12.345 | 345.789,239 | 24.987 | 234.567,893
en_GB: 12,345 | 345,789.239 | 24,987 | 234,567.893
```

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/i18n/format/numberFormat.html
(Bentley, TutorialAtOracle Code Examples, 2013) InternationalisationDemo.java

Format int and double numbers with printf. (fix*) Note, for doubles, the precision with digits after the decimal point is retained.

```
private static void demoNumberFormatingWithPrintf() {
  Locale[] locales =
      { new Locale("fr", "FR"), new Locale("de", "DE"), new Locale("en", "GB") };

  Integer population = new Integer(12345);
  Double height = new Double(345789.2392345);
  int fleetQuantity = 24987;
  double distance = 234567.893483;

  NumberFormat numberFormat = null;
  for (Locale localeLoop : locales) {
    numberFormat = NumberFormat.getNumberInstance(localeLoop);
    System.out.printf(localeLoop, "%s: %,d | %,f | %,d | %,f%n",
        localeLoop.toString(),
        population,
        height,
        fleetQuantity,
        distance
    );
  }
}

// Output
fr_FR: 12 345 | 345 789,239235 | 24 987 | 234 567,893483
de_DE: 12.345 | 345.789,239235 | 24.987 | 234.567,893483
en_GB: 12,345 | 345,789.239235 | 24,987 | 234,567.893483
```

(Bentley, TutorialAtOracle Code Examples, 2013) InternationalisationDemo.java
(Bentley, Java Reference - Language - Printf Style Formatting.docx, 2013)

Currencies

Currency Codes.

```
 "EUR", "USD", "AUD", ...
```

http://en.wikipedia.org/wiki/ISO_4217#Active_codes

Display currencies with NumberFormat.getCurrencyInstance() and the Currency class (for symbols and codes only). In theory the code looks like the following ...

```
private static void demoCurrencyFormattingWithNumberFormat() {
  Locale[] locales =
      { new Locale("fr", "FR"), new Locale("de", "DE"), new Locale("en", "GB"),
          new Locale("en", "US") };

  Double price = new Double(345789.2392345);

  NumberFormat currencyFormat = null;
  Currency currency = null;
  price = price * 1234.123456789101112131415;
  for (Locale localeLoop : locales) {
    currencyFormat = NumberFormat.getCurrencyInstance(localeLoop);
    currency = Currency.getInstance(localeLoop);
    System.out.printf(localeLoop, "%s: %s | %s | %s%n",
        localeLoop.toString(),
        currency.getCurrencyCode(),
        currency.getSymbol(localeLoop), // Must specify locale else code returned
        currencyFormat.format(price)
    );
  }
  System.out.println("price: " + price.toString());
}

// Output
fr_FR: EUR | € | 345 789,24 €
```

```
de_DE: EUR | € | 345.789,24 €
en_GB: GBP | £ | £345,789.24
en_US: USD | $ | $345,789.24
price: 4.267466112445546E8
```

There is no printf equivalent for displaying locale sensitive currency values or symbols (but you could concatenate Currency class symbols with a formatted printf number.

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/i18n/format/numberFormat.html
(Bentley, TutorialAtOracle Code Examples, 2013) InternationalisationDemo.java

... However, you should use BigDecimal for storing currency values. Therefore in practice, you should do something like ...

```
private static void demoCurrencyFormattingUsingBigDecimal() {
  Locale[] locales =
      { new Locale("fr", "FR"), new Locale("de", "DE"), new Locale("en", "GB"),
        new Locale("en", "US") };

  BigDecimal price = new BigDecimal(345789.2392345);

  //Used to perform rounding before conversion to a double.
  BigDecimal priceDisplay = null;

  NumberFormat currencyFormat = null;
  Currency currency = null;

  // Perform error prone calculations due to rounding error
  price = price.multiply(new BigDecimal(1234.123456789101112131415));
  for (Locale localeLoop : locales) {
    currencyFormat = NumberFormat.getCurrencyInstance(localeLoop);
    currency = Currency.getInstance(localeLoop);
    priceDisplay = price.setScale(2, BigDecimal.ROUND_HALF_EVEN);
    System.out.printf(localeLoop, "%s: %s | %s | %s%n",
        localeLoop.toString(),
        currency.getCurrencyCode(),
        currency.getSymbol(localeLoop), // Must specify locale else code returned
        currencyFormat.format(priceDisplay.doubleValue())
    );
  }
  System.out.println("price: " + price.toString());
  System.out.println("priceDisplay: " + priceDisplay.toString());
}

// Output
fr_FR: EUR | € | 426 992 617,19 €
de_DE: EUR | € | 426.992.617,19 €
en_GB: GBP | £ | £426,992,617.19
en_US: USD | $ | $426,992,617.19
price: 426992617.19331750015734383250848471720844540255868082567758392542600631713867187 5
priceDisplay: 426992617.19
```

http://stackoverflow.com/questions/1359817/using-bigdecimal-to-work-with-currencies &
(Bentley, TutorialAtOracle Code Examples, 2013) InternationalisationDemo.java

Percentage

Percentage formatting with NumberFormat.

```
private static void demoPercentageFormattingWithNumberFormat() {
  Locale[] locales =
      { new Locale("fr", "FR"), new Locale("de", "DE"), new Locale("en", "GB"),
        new Locale("en", "US") };

  Double portion = new Double(0.752392345);

  NumberFormat percentageFormat = null;
  for (Locale localeLoop : locales) {
    percentageFormat = NumberFormat.getPercentInstance(localeLoop);
```

```
    System.out.printf(localeLoop, "%s: %s%n",
        localeLoop.toString(),
        percentageFormat.format(portion)
    );
  }
}

// Ouput
fr_FR: 75 %
de_DE: 75%
en_GB: 75%
en_US: 75%
```

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/i18n/format/numberFormat.html
(Bentley, TutorialAtOracle Code Examples, 2013) InternationalisationDemo.java

Custom Formatting

Custom number formatting example (a currency example), with the default locale

```
private static void demoCustomNumberFormattingWithDecimalFormatDefaultLocale() {

  BigDecimal price = new BigDecimal(345789.2392345);
  BigDecimal priceDisplay = null;

  priceDisplay = price.setScale(2, BigDecimal.ROUND_HALF_EVEN);

  DecimalFormat formatter = new DecimalFormat("¤¤ ¤ ###,###.##");

  System.out.printf("%s: %s%n", Locale.getDefault().toString(),
      formatter.format(priceDisplay.doubleValue()));
}
```

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/i18n/format/decimalFormat.html
(Bentley, TutorialAtOracle Code Examples, 2013) InternationalisationDemo.java

Custom number formatting example (a currency example), with multiple locales.

```
private static void demoCustomNumberFormattingWithDecimalFormat() {
  Locale[] locales =
      { new Locale("fr", "FR"), new Locale("de", "DE"), new Locale("en", "GB") };

  BigDecimal price = new BigDecimal(345789.2392345);
  BigDecimal priceDisplay = null;

  NumberFormat nf = null;
  DecimalFormat formatter = null;
  priceDisplay = price.setScale(2, BigDecimal.ROUND_HALF_EVEN);
  for (Locale localeLoop : locales) {
    // For localisation cast a localised NumberFormat to a DecimalFormat object
    nf = NumberFormat.getNumberInstance(localeLoop);
    formatter = (DecimalFormat) nf;

    formatter.applyPattern("¤¤ ¤ ###,###.##");
    System.out.printf(localeLoop, "%s: %s%n", localeLoop.toString(),
        formatter.format(priceDisplay.doubleValue()));
  }
}

// Output
fr_FR: EUR € 345 789,24
de_DE: EUR € 345.789,24
en_GB: GBP £ 345,789.24
```

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/i18n/format/decimalFormat.html
(Bentley, TutorialAtOracle Code Examples, 2013) InternationalisationDemo.java

Custom Number format Syntax (for use with DecimalFormat):

Backus -Naur Form ...

```
pattern    := subpattern{;subpattern}
subpattern := {prefix}integer{.fraction}{suffix}
prefix     := '\\u0000'..'\\uFFFD' - specialCharacters
suffix     := '\\u0000'..'\\uFFFD' - specialCharacters
integer    := '#'* '0'* '0'
fraction   := '0'* '#'*
```

Meaning of Backus-Naur Form above ...

Notation	Description
X*	0 or more instances of X
(X | Y)	either X or Y
X..Y	any character from X up to Y, inclusive
S - T	characters in S, except those in T
{X}	X is optional

Meaning of symbols ...

Symbol	Description
0	a digit
#	a digit, zero shows as absent
.	placeholder for decimal separator
,	placeholder for grouping separator
E	separates mantissa and exponent for exponential formats
;	separates formats
-	default negative prefix
%	multiply by 100 and show as percentage
?	multiply by 1000 and show as per mille
¤	currency sign; replaced by currency symbol; if doubled, replaced by international currency symbol; if present in a pattern, the monetary decimal separator is used instead of the decimal separator
X	any other characters can be used in the prefix or suffix
'	used to quote special characters in a prefix or suffix

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/i18n/format/decimalFormat.html

To alter the format symbols use DecimalFormatSymbols.

```
private static void demoCustomNumberFormattingAlteringTheFormatSymbols() {
  DecimalFormatSymbols symbols = new DecimalFormatSymbols();
```

```
  symbols.setGroupingSeparator(' ');

  DecimalFormat formatter = new DecimalFormat("##,####", symbols);
  long cardNumber = 4024007170584121L;

  System.out.printf("Card Number: %s%n",formatter.format(cardNumber));
}

// Output
Card Number: 4024 0071 7058 4121
```

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/i18n/format/decimalFormat.html
(Bentley, TutorialAtOracle Code Examples, 2013) InternationalisationDemo.java

Dates and Times

Overview

Which Java date and time APIs to use:

- Java7 and earlier: use `java.sql.Date` when you use libraries that depend on it (like JDBC). Use `java.util.Date` (and related) when you call libraries that depend on it. Otherwise use JODA.
- Java8 and later: use `java.time`, a native api based on JODA.

Note at http://www.joda.org/joda-time/ it is stated

> ... from Java SE 8 onwards, users are asked to migrate to java.time (JSR-310) - a core part of the JDK which replaces this project.

Stackoverflow > java.util.Date vs java.sql.Date. Answer by gustafc, updated 2015-05-29 08:17,
http://stackoverflow.com/a/2306756
http://www.joda.org/joda-time/

Handle DateTimes with Calendar, GregorianCalendar and Date classes. Date classes has many methods that are deprecated, so use GregorianCalendar (a subclass of Calendar) by preference. If you run into problems or need sophisticated DateTime handling try the third party DateTime class Joda.

Note Calendar can handle TimeZone translations but Date cannot (without using hacky deprecated methods).

(joda.org, 2013)

Format datetimes using predefined or custom formats. Your options are:

- Predefined: DateFormat, Printf Style.
- Custom: SimpleDateFormat (recommended), Printf Style.

See (Oracle, 2017) http://docs.oracle.com/javase/tutorial/i18n/format/dateFormat.html
(Oracle, 2017) http://docs.oracle.com/javase/tutorial/i18n/format/simpleDateFormat.html
(Bentley, Java Reference - Language - Printf Style Formatting.docx, 2013)
(Bentley, TutorialAtOracle Code Examples, 2013) DateTimeDemo.java

You'll generally avoid using Predefined DateTime Formatting with DateFormat as it doesn't support ISO 8601, the international standard for Date and Time representation. Use, instead, custom formatting with SimpleDateFormat (recommended) or Printf style formatting. SimpleDateFormat can return daylight savings full timezone name automatically. This makes it easier to use than printf style formatting (which would require additional use of the TimeZone class).

```
public static void nowCustomFormatSimpleDateFormatSimpleExample() {
  Calendar nowCalendar = GregorianCalendar.getInstance();

  // SimpleDateFormat can return daylight savings timezone name automatically.
  // in 'z' and 'zzzz'. E.g. 'CDT' and Central Daylight Time.
  SimpleDateFormat sdf = new SimpleDateFormat("yyyy-MM(MMM)-dd HH:mm:ss | Z, z, zzzz");
  sdf.setTimeZone(TimeZone.getTimeZone("America/Chicago"));

  Date nowDate = nowCalendar.getTime();
  System.out.println(sdf.format(nowDate));
}

// Output
2013-08(Aug)-15 01:37:23 | -0500, CDT, Central Daylight Time
```

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/i18n/format/simpleDateFormat.html
(Bentley, TutorialAtOracle Code Examples, 2013) DateTimeDemo.java

Get now string example.

```
static public String getNowString() {

    // SimpleDateFormat can return daylight savings timezone name automatically.
    // in 'z' and 'zzzz'. E.g. 'CDT' and Central Daylight Time.
    SimpleDateFormat sdf = new SimpleDateFormat("yyyy-MM-dd | HH:mm:ss | Z");
    sdf.setTimeZone(TimeZone.getTimeZone("Australia/Sydney"));

    Calendar nowCalendar = GregorianCalendar.getInstance();
    Date nowDate = nowCalendar.getTime();

    return sdf.format(nowDate);
}

// Output
2014-06-17 | 05:15:46 | +1000
```

C:\Users\John\Documents\Sda\Code\Android\Examples\Training\MyFirstApp02\standardapplibrary\src\main\java\au\com\softmake\standardapplibrary\java\DateTimeHelper.java

Date() to Calendar() conversions

The Java 6 DateTime implementation is so schizophrenic that you might find yourself needing to convert a DateTime back and forth between a Date() object and a Calendar() object.

```
private static void DateToCalendarConversions() {
  SimpleDateFormat sdf = new SimpleDateFormat("yyyy-MM(MMM)-dd HH:mm");

  Date nowDate = new Date();
  System.out.printf("%s%n", sdf.format(nowDate));

  // Convert Date to Calendar Object for any manipulation as follows.
  Calendar dateTimeCalendar = GregorianCalendar.getInstance();
  dateTimeCalendar.setTime(nowDate);
  dateTimeCalendar.set(Calendar.MONDAY, Calendar.JANUARY);

  // Output Calendar object with printf
  String printfDateTimeFormatString = "%1$tY-%1$tm(%1tb)-%1$td %1$tR%n";
  System.out.printf(printfDateTimeFormatString, dateTimeCalendar);

  // Convert Calendar back to Date Object for use with sdf.format
  Date dateTimeDate = dateTimeCalendar.getTime();
  System.out.printf("%s%n", sdf.format(dateTimeDate));
}
```

(Bentley, TutorialAtOracle Code Examples, 2013) DateTimeDemo.java

Formating Dates and Times

Predefined DateTime Formatting with DateFormat

Format datetimes with predefined formats using DateFormat. This is Locale sensitive.

```
private static void demoDateTimeFormattingPredefinedSimple() {
  Date nowDate = new Date();

  DateFormat dateDateFormat = DateFormat.getDateInstance(DateFormat.DEFAULT);
  DateFormat timeDateFormat = DateFormat.getTimeInstance(DateFormat.DEFAULT);
  DateFormat dateTimeDateFormat =
      DateFormat.getDateTimeInstance(DateFormat.DEFAULT, DateFormat.DEFAULT);

  // DateFormat.format accepts Date objects not Calendar objects.
  System.out.printf("%s: %12s | %10s | %s%n", Locale.getDefault().toString(),
      dateDateFormat.format(nowDate), timeDateFormat.format(nowDate),
      dateTimeDateFormat.format(nowDate));
}

// Output
en_AU:   14/08/2013 | 5:19:12 PM | 14/08/2013 5:19:12 PM
```

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/i18n/format/dateFormat.html
(Bentley, TutorialAtOracle Code Examples, 2013) DateTimeDemo.java

DateFormat takes a Date object but you can convert a Calendar object to a Date Object if needed. You can also format datetimes for different Locales.

```
private static void demoDateTimeFormattingPredefinedMultipleLocales () {
  Calendar nowCalendar = GregorianCalendar.getInstance();

  // Create DateTime instance using Date, without using Calendar
//    Date nowDate = new Date();

  // Demo conversion from Calendar (the newer class) to
  // Date (the class with many deprecated methods)
  Date nowDate = nowCalendar.getTime();

  Locale[] locales =
    { new Locale("fr", "FR"), new Locale("de", "DE"), new Locale("en", "GB"),
        new Locale("en", "US"), new Locale("en", "AU") };

  DateFormat dateDateFormat = null;
  DateFormat timeDateFormat = null;
  DateFormat dateTimeDateFormat = null;

  for (Locale localeLoop : locales){
    dateDateFormat = DateFormat.getDateInstance(DateFormat.DEFAULT, localeLoop);
    timeDateFormat = DateFormat.getTimeInstance(DateFormat.DEFAULT, localeLoop);
    dateTimeDateFormat =
        DateFormat.getDateTimeInstance(DateFormat.DEFAULT, DateFormat.DEFAULT, localeLoop);

    // DateFormat.format accepts Date objects not Calendar objects.
    System.out.printf("%s: %12s | %10s | %s%n", localeLoop.toString(),
        dateDateFormat.format(nowDate),
        timeDateFormat.format(nowDate),
        dateTimeDateFormat.format(nowDate)
    );
  }
}

// Output
fr_FR: 14 août 2013 |   17:05:32 | 14 août 2013 17:05:32
de_DE:   14.08.2013 |   17:05:32 | 14.08.2013 17:05:32
en_GB:  14-Aug-2013 |   17:05:32 | 14-Aug-2013 17:05:32
en_US: Aug 14, 2013 | 5:05:32 PM | Aug 14, 2013 5:05:32 PM
en_AU:   14/08/2013 | 5:05:32 PM | 14/08/2013 5:05:32 PM
```

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/i18n/format/dateFormat.html
(Bentley, TutorialAtOracle Code Examples, 2013) DateTimeDemo.java

DateFormat predefined styles.

```
dateTimeDateFormat =
        DateFormat.getDateTimeInstance(DateFormat.DEFAULT, DateFormat.DEFAULT, localeLoop);

...
dateTimeDateFormat.format(nowDate)
```

Style	fr_FR	de_DE	en_GB	en_US	en_AU (current locale)
DEFAULT	14 août 2013 17:49:32	14.08.2013 17:49:32	14-Aug-2013 17:49:32	Aug 14, 2013 5:49:32 PM	14/08/2013 5:49:32 PM
SHORT	14/08/13 17:49	14.08.13 17:49	14/08/13 17:49	8/14/13 5:49 PM	14/08/13 5:49 PM
MEDIUM	14 août 2013 17:49:32	14.08.2013 17:49:32	14-Aug-2013 17:49:32	Aug 14, 2013 5:49:32 PM	14/08/2013 5:49:32 PM
LONG	14 août 2013 17:49:32 EST	14. August 2013 17:49:32 EST	14 August 2013 17:49:32 EST	August 14, 2013 5:49:32 PM EST	14 August 2013 5:49:32 PM
FULL	mercredi 14 août 2013 17 h 49 EST	Mittwoch, 14. August 2013 17:49 Uhr EST	Wednesday, 14 August 2013 17:49:32 o'clock EST	Wednesday, August 14, 2013 5:49:32 PM EST	Wednesday, 14 August 2013 5:49:32 PM EST

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/i18n/format/dateFormat.html
(Bentley, TutorialAtOracle Code Examples, 2013) DateTimeDemo.java

Custom DateTime Formatting

With SimpleDateFormat

SimpleDateFormat Overview

Custom DateTime Formatting with SimpleDateFormat, simple example. You must pass a Date object to the SimpleDateFormat.format() method, but you can retrieve this from a Calender object with getTime().

```
public static void nowCustomFormatSimpleDateFormatSimpleExample() {
  Calendar nowCalendar = GregorianCalendar.getInstance();

  // SimpleDateFormat can return daylight savings timezone name automatically.
  // in 'z' and 'zzzz'. E.g. 'CDT' and Central Daylight Time.
  SimpleDateFormat sdf = new SimpleDateFormat("yyyy-MM(MMM)-dd HH:mm:ss | Z, z, zzzz");

  Date nowDate = nowCalendar.getTime();
  System.out.println(sdf.format(nowDate));
}

// Output
2013-08(Aug)-15 16:28:50 | +1000, EST, Eastern Standard Time (New South Wales)
```

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/i18n/format/simpleDateFormat.html
(Bentley, TutorialAtOracle Code Examples, 2013) DateTimeDemo.java

Custom DateTime Formatting with SimpleDateFormat, full example. You must pass a Date object to the SimpleDateFormat.format() method, but you can retrieve this from a Calender object with getTime().

```
public static void nowCustomFormatSimpleDateFormatFullExample () {
  Calendar nowCalendar = GregorianCalendar.getInstance();
  String sdfPattern = "yyyy-MM(MMM)-dd HH:mm:ss | Z, z, zzzz";

  Locale[] locales =
      { Locale.getDefault(), new Locale("fr", "FR"), new Locale("es", "ES"),
        new Locale("en", "GB"), new Locale("en", "US"), new Locale("en", "AU") };

  TimeZone[] timezones =
      { TimeZone.getDefault(), TimeZone.getTimeZone("Australia/Perth"),
        TimeZone.getTimeZone("UTC"), TimeZone.getTimeZone("America/Chicago"),
        TimeZone.getTimeZone("Pacific/Fiji") };

  for (Locale localeLoop : locales) {
    System.out.printf("%s | %s%n", localeLoop.toString(), localeLoop.getDisplayName());
    SimpleDateFormat sdf = new SimpleDateFormat(sdfPattern, localeLoop);

    for (TimeZone timeZoneLoop : timezones) {
      sdf.setTimeZone(timeZoneLoop);
      System.out.printf("%16s | ", timeZoneLoop.getID());

      // SimpleDateFormat can return daylight savings timezone name automatically.
      // in 'z' and 'zzzz'. E.g. 'CDT' and Central Daylight Time.
      System.out.println(sdf.format(nowCalendar.getTime()));
    }
    System.out.println();
  }
}

// Output
en_AU | English (Australia)
Australia/Sydney | 2013-08(Aug)-15 16:19:49 | +1000, EST, Eastern Standard Time (New South
Wales)
 Australia/Perth | 2013-08(Aug)-15 14:19:49 | +0800, WST, Western Standard Time (Australia)
             UTC | 2013-08(Aug)-15 06:19:49 | +0000, UTC, Coordinated Universal Time
 America/Chicago | 2013-08(Aug)-15 01:19:49 | -0500, CDT, Central Daylight Time
    Pacific/Fiji | 2013-08(Aug)-15 18:19:49 | +1200, FJT, Fiji Time

fr_FR | French (France)
Australia/Sydney | 2013-08(août)-15 16:19:49 | +1000, EST, Heure normale de l'Est (Nouvelle-
Galles du Sud)
 Australia/Perth | 2013-08(août)-15 14:19:49 | +0800, WST, Heure normale de l'Ouest
(Australie)
             UTC | 2013-08(août)-15 06:19:49 | +0000, UTC, Temps universel coordonné
 America/Chicago | 2013-08(août)-15 01:19:49 | -0500, CDT, Heure avancée du Centre
    Pacific/Fiji | 2013-08(août)-15 18:19:49 | +1200, FJT, Heure de Fidji
...
```

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/i18n/format/simpleDateFormat.html
(Bentley, TutorialAtOracle Code Examples, 2013) DateTimeDemo.java

SimpleDateFormat Pattern Syntax

Pattern characters.

Letter	Date or Time Component	Presentation	Examples
G	Era designator	Text	AD
y	Year	Year	1996; 96
M	Month in year	Month	July; Jul; 07
w	Week in year	Number	27
W	Week in month	Number	2

D	Day in year	Number	189
d	Day in month	Number	10
F	Day of week in month	Number	2
E	Day in week	Text	Tuesday; Tue
a	Am/pm marker	Text	PM
H	Hour in day (0-23)	Number	0
k	Hour in day (1-24)	Number	24
K	Hour in am/pm (0-11)	Number	0
h	Hour in am/pm (1-12)	Number	12
m	Minute in hour	Number	30
s	Second in minute	Number	55
S	Millisecond	Number	978
z	Time zone	General time zone	Pacific Standard Time; PST; GMT-08:00
Z	Time zone	RFC 822 time zone	-0800

(Oracle, 2011) http://docs.oracle.com/javase/6/docs/api/java/text/SimpleDateFormat.html *&*
(Oracle, 2017) http://docs.oracle.com/javase/tutorial/i18n/format/simpleDateFormat.html

The number of character symbols you specify determines the format, according to the following.

Presentation	**Number of Symbols**	**Result**	**Examples**
Text	1 - 3	abbreviated form, if one exists	EEE=Fri
Text	>= 4	full form	MMMM=August EEEE=Friday
Number	minimum number of digits is required	shorter numbers are padded with zeros (for a year, if the count of 'y' is 2, then the year is truncated to 2 digits)	hh=03 yy=13
Month	3	text form	MMM=Aug
Month	1 - 2	number form	MM=08 M=8
TimeZone, RFC 822	1	Text form, short	Z=-0800
TimeZone, General	4	Text form, long	zzzz=Eastern Standard Time (New South Wales)
TimeZone, General	1-3	Text from, three letter abbreviation	zzz=EST (or " GMT-08:00 ") zz=EST (or " GMT-08:00 ") z=EST (or " GMT-08:00 ")

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/i18n/format/simpleDateFormat.html

Non character text, other than above, a treated literally.

```
"yyyy!!" -> 2013!!
```

Escape characters by surrounding them in a single quote, '.

```
"yyyy-MM 'yyyy-mm'" -> 2013-08 yyyy-mm
"yyyy-MM 'Cool'" - > 2013-08 Cool
```

Escape a single quote by doubling the single quote, ''.

```
"yyyy-MM ''" -> "2013-08 '"
"yyyy-MM '''Cool'''" -> 2013-08 'Cool'
```

SimpleDateFormat Pattern Syntax Examples

SimpleDateFormat Pattern Syntax Examples

Description	SimpleDateFormat Pattern	Output
ISO 8601	yyyy-MM-dd HH:mm:ss.S Z	2013-08-16 15:38:48.35 +1000
ISO 8601	yyyy-MM-dd	2013-08-16
ISO 8601	HH:mm:ss	15:38:48
JLB Format	yyyy-MM(MMM)-dd HH:mm:ss.S Z z zzzz	2013-08(Aug)-16 15:38:48.35 +1000 EST Eastern Standard Time (New South Wales)
JLB Format	yyyy-MM(MMM)-dd	2013-08(Aug)-16
Year Era	yyyy G	2013 AD
Colloquial	EEE, dd MMM, 'at' h:mm a	Fri, 16 Aug, at 3:38 PM

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/i18n/format/simpleDateFormat.html &
(Bentley, TutorialAtOracle Code Examples, 2013) DateTimeDemo.java

Alter the SimpleDateFormat character output symbols

Create your own datetime symbols with DateFormatSymbols.

```
private static void nowCustomFormatDateFormatSymbols(){
  Calendar nowCalendar = GregorianCalendar.getInstance();
  Date nowDate = nowCalendar.getTime();

  DateFormatSymbols symbols = new DateFormatSymbols();
  String[] fucknDays = { "", "SunFuckn", "MonFuckn", "TueFuckn", "WedFuckn", "ThuFuck",
      "FriFuckn", "SatFuckn" };
  symbols.setWeekdays(fucknDays); // Long format weekdays
  SimpleDateFormat sdf = new SimpleDateFormat("EEEE", symbols);

  System.out.printf("%s%n", sdf.format(nowDate));
}

// Output (when run on a Friday)
FriFuckn [Normally "Friday"]
```

Setter Method	Example of a Symbol the Method Modifies
setAmPmStrings	PM
setEras	AD
setMonths	December
setShortMonths	Dec
setShortWeekdays	Tue
setWeekdays	Tuesday

setZoneStrings	PST

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/i18n/format/dateFormatSymbols.html &
(Bentley, TutorialAtOracle Code Examples, 2013) DateTimeDemo.java

With Printf Style Formatting

See (Bentley, Java Reference - Language - Printf Style Formatting.docx, 2013) for syntax. Custom DateTime Formatting with printf style formatting, simple example. You must use Calendar (and best with GregorianCalendar), or Date for that which you pass to printf.

```
public static void nowCustomFormatPrintfSimpleExample() {
  Calendar nowCalendar = GregorianCalendar.getInstance();
  String printfDateTimeFormatString = "%1$tY-%1$tm(%1tb)-%1$td %1$tT UTC%1$tz %1$tZ";
  System.out.printf(printfDateTimeFormatString, nowCalendar);
}

// Output
2013-08(Aug)-15 08:16:28 UTC+1000 EST
```

(Bentley, Java Reference - Language - Printf Style Formatting.docx, 2013)
(Bentley, TutorialAtOracle Code Examples - HelloWorld, 2013) DateTimeDemo.java

Custom DateTime Formatting with printf style formatting, full example. You can use Calendar (and best with GregorianCalendar), or Date for that which you pass to printf.

```
public static void nowCustomFormatPrintfFullExample() {
  Calendar nowCalendar = GregorianCalendar.getInstance();
  String printfDateTimeFormatString = "%1$tY-%1$tm(%1tb)-%1$td %1$tT UTC%1$tz %1$tZ";

  Locale[] locales =
      { Locale.getDefault(), new Locale("fr", "FR"), new Locale("es", "ES"),
        new Locale("en", "GB"), new Locale("en", "US"), new Locale("en", "AU") };

  TimeZone[] timeZones =
      { TimeZone.getDefault(), TimeZone.getTimeZone("Australia/Perth"),
        TimeZone.getTimeZone("UTC"), TimeZone.getTimeZone("America/Chicago"),
        TimeZone.getTimeZone("Pacific/Fiji") };

  for (Locale localeLoop : locales) {
    System.out.printf("%s | %s%n", localeLoop.toString(), localeLoop.getDisplayName());
    for (TimeZone timeZoneLoop : timeZones) {
      System.out.printf("%16s | ", timeZoneLoop.getID());
      System.out.printf(localeLoop, printfDateTimeFormatString, nowCalendar);
      boolean inDaylightTime = timeZoneLoop.inDaylightTime(nowCalendar.getTime());
      System.out.printf(" | %s%n",
          timeZoneLoop.getDisplayName(inDaylightTime, TimeZone.LONG, localeLoop));
    }
    System.out.println();
  }
} // nowCustomFormatPrintf

// Output
en_AU | English (Australia)
Australia/Sydney | 2013-08(Aug)-15 08:00:49 UTC+1000 EST | Eastern Standard Time (New South
Wales)
 Australia/Perth | 2013-08(Aug)-15 06:00:49 UTC+0800 WST | Western Standard Time (Australia)
             UTC | 2013-08(Aug)-14 22:00:49 UTC+0000 UTC | Coordinated Universal Time
 America/Chicago | 2013-08(Aug)-14 17:00:49 UTC-0600 CDT | Central Daylight Time
    Pacific/Fiji | 2013-08(Aug)-15 10:00:49 UTC+1200 FJT | Fiji Time

fr_FR | French (France)
Australia/Sydney | 2013-08(août)-15 08:00:49 UTC+1000 EST | Heure normale de l'Est
(Nouvelle-Galles du Sud)
 Australia/Perth | 2013-08(août)-15 06:00:49 UTC+0800 WST | Heure normale de l'Ouest
(Australie)
             UTC | 2013-08(août)-14 22:00:49 UTC+0000 UTC | Temps universel coordonné
 America/Chicago | 2013-08(août)-14 17:00:49 UTC-0600 CDT | Heure avancée du Centre
```

```
    Pacific/Fiji | 2013-08(août)-15 10:00:49 UTC+1200 FJT | Heure de Fidji

...
```

Component Description	Examples	Code Example
Locale language and region code	en_AU, fr_FR	localeLoop.toString()
Locale display name	English (Australia), French (France)	localeLoop.getDisplayName()
TimeZone ID	Australia/Sydney, UTC	timeZoneLoop.getID()
Printf formatting	2013-08(Aug)-15 08:00:49 UTC+1000 EST 2013-08(août)-15 06:00:49 UTC+0800 WST	System.out.printf(localeLoop, "%1$tY-%1$tm(%1tb)-%1$td %1$tT UTC%1$tz %1$tZ", nowCalendar);
TimeZone Long Name, Daylight sensitive	Eastern Standard Time (New South Wales) Heure normale de l'Ouest (Australie)	boolean inDaylightTime = timeZoneLoop.inDaylightTime(nowCalendar.getTime()); System.out.printf(" | %s%n", timeZoneLoop.getDisplayName(inDaylightTime, TimeZone.LONG, localeLoop));

(Bentley, Java Reference - Language - Printf Style Formatting.docx, 2013)
(Bentley, TutorialAtOracle Code Examples, 2013) DateTimeDemo.java

TimeZones

List of TimeZone IDs.

```
public static void listTimeZoneAvailableIDs() {
  String[] timeZoneIDs = TimeZone.getAvailableIDs();

  for (String id : timeZoneIDs) {
    System.out.println(id);
  }
}

// Output
Etc/GMT+12
Etc/GMT+11
Pacific/Midway
Pacific/Niue
...
UTC
...
Etc/GMT-9
...
Australia/ACT
Australia/Brisbane
Australia/Canberra
Australia/Hobart
Australia/Melbourne
Australia/NSW
Australia/Queensland
Australia/Sydney
Australia/Tasmania
Australia/Victoria
```

(Bentley, TutorialAtOracle Code Examples, 2013) DateTimeDemo.java

Return the same DateTime in different timezones. Use TimeZone and SimpleDateFormat.

```
public static void nowTimeZoneTranslation() {
  Calendar nowCalendar = GregorianCalendar.getInstance();
  Date nowDate = nowCalendar.getTime();
  SimpleDateFormat sdf = new SimpleDateFormat("yyyy-MM(MMM)-dd HH:mm:ss | Z, z, zzzz");

  TimeZone[] timezones =
      { TimeZone.getDefault(), TimeZone.getTimeZone("Australia/Perth"),
          TimeZone.getTimeZone("UTC"), TimeZone.getTimeZone("America/Chicago"),
          TimeZone.getTimeZone("Pacific/Fiji") };

  for (TimeZone timeZoneLoop : timezones) {
    sdf.setTimeZone(timeZoneLoop);
    System.out.printf("%16s | ", timeZoneLoop.getID());
    System.out.printf("%s%n", sdf.format(nowDate));
  }
}

// Output
Australia/Sydney | 2013-08(Aug)-16 23:16:27 | +1000, EST, Eastern Standard Time (New South
Wales)
 Australia/Perth | 2013-08(Aug)-16 21:16:27 | +0800, WST, Western Standard Time (Australia)
             UTC | 2013-08(Aug)-16 13:16:27 | +0000, UTC, Coordinated Universal Time
 America/Chicago | 2013-08(Aug)-16 08:16:27 | -0500, CDT, Central Daylight Time
    Pacific/Fiji | 2013-08(Aug)-17 01:16:27 | +1200, FJT, Fiji TimeParsing Dates and Times
```

(Bentley, TutorialAtOracle Code Examples, 2013) DateTimeDemo.java

Parsing Dates and Times

To parse dates and times use SimpleDateFormat.

```
private static void parseDateTimes() {
  SimpleDateFormat sdfInput = new SimpleDateFormat("yyyy-MM-dd HH:mm");
  SimpleDateFormat sdfOutput = new SimpleDateFormat("EEE, dd MMM yyyy, 'at' h:mm a");
  Date dateTimeDate = null;

  try {
    dateTimeDate = sdfInput.parse("2013-06-30 21:15");
    System.out.printf("%s%n", sdfOutput.format(dateTimeDate));
  } catch (ParseException e) {
    e.printStackTrace();
  }
}

// Output
Sun, 30 Jun 2013, at 9:15 PM
```

(Bentley, TutorialAtOracle Code Examples, 2013) DateTimeDemo.java

Working with Text

Dealing with compound messages

Deal with compound messages by joining a RescourceBundle pattern string, MessageFormat, and message arguments.

```
// Create resource bundle property files
// Message.properties
pattern = At {2,time,short} on {2,date,long}, we detected {1,number,integer} spaceships on
the planet {0}.
planet = Mars

// Message_de_DE.properties
pattern = Um {2,time,short} {2,date,long} haben wir {1,number,integer} Raumschiffe auf dem
Planeten {0} entdeckt.
```

```
planet = Mars

private static void demoCompoundMessages(){
  Locale[] locales = { Locale.getDefault(), new Locale ("de", "DE") };

  for (Locale localeLoop : locales) {

    // 1. Get resource bundle from property file
    ResourceBundle messageRB = ResourceBundle.getBundle("helloworld.resources.Message",
localeLoop);

    // 2. Specify message arguments, some of which from the property file.
    Object[] messageArguments = {
        messageRB.getString("planet"),
        new Integer(7),
        new Date()
    };

    // 3. Create a MessageFormat from a template string.
    MessageFormat mf = new MessageFormat(messageRB.getString("pattern"), localeLoop);

    System.out.printf("%s: ", localeLoop.toString());

    // 4. Get output string by merging variable values with template string.
    System.out.println(mf.format(messageArguments));
  }
}

// Output
en_AU: At 10:12 AM on 18 August 2013, we detected 7 spaceships on the planet Mars.
de_DE: Um 10:12 18. August 2013 haben wir 7 Raumschiffe auf dem Planeten Mars entdeckt.
```

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/i18n/format/messageFormat.html
(Bentley, TutorialAtOracle Code Examples, 2013) InternationalisationDemo.java

Handling Plurals

Handling plurals example.

```
// Create resource bundle property files
// FileChoice.properties
pattern = There {0} on {1}.
noFiles = are no files
oneFile = is one file
multipleFiles = are {2} files

// FileChoice fr FR.properties
pattern = Il {0} sur {1}.
noFiles = n'y a pas de fichiers
oneFile = y a un fichier
multipleFiles = y a {2} fichiers

/**
 * Numbers correspond to the steps at
 * http://docs.oracle.com/javase/tutorial/i18n/format/choiceFormat.html
 */
private static void demoHandlingPlurals() {
  Locale[] locales = { Locale.getDefault(), new Locale ("fr", "FR") };

  for (Locale localeLoop : locales) {
    System.out.printf("%s:%n", localeLoop.toString());

    // 1. Create resource bundle property files

    // 2. Get resource bundle from property file
    ResourceBundle fileChoiceRB =
ResourceBundle.getBundle("helloworld.resources.FileChoice", localeLoop);

    // 3. Create a MessageFormat from a template string.
    MessageFormat mf = new MessageFormat(fileChoiceRB.getString("pattern"), localeLoop);

    // 4. Choice formatter
    double[] fileQuantity = {0,1,2};
```

```
    String[] fileStrings = {
        fileChoiceRB.getString("noFiles"),
        fileChoiceRB.getString("oneFile"),
        fileChoiceRB.getString("multipleFiles")
    };
    ChoiceFormat cf = new ChoiceFormat(fileQuantity, fileStrings);

    // 5. Patten already assigned

    // 6. Join the Choice Format with the Message Format
    // The setFormats method assigns Format objects to the arguments in the message pattern.
    Format[] formats = {cf, null, NumberFormat.getInstance()};
    mf.setFormats(formats);

    // 7. Create Message arguments.
    // 7A. Set messageArgument {1}
    Object[] messageArguments = {null, "Disk X", null};

    for (int numFiles = 0; numFiles <= 3; numFiles++) {
        messageArguments[0] = new Integer(numFiles);
        messageArguments[2] = new Integer(numFiles);

        // Example, for num files 3
        /* messageArgument   formats
         *  {0} 3             cf( 3 ) -> "are {2} files"
         *  {1} "Disk X"
         *  {2} 3             "are {2} files" -> "are 3 files"
         */

        // 7B. Insert messageArguments into MessageFormatter to get result.
        System.out.println(mf.format(messageArguments));
    }
    System.out.println();
  }
}

// Output
en AU:
There are no files on Disk X.
There is one file on Disk X.
There are 2 files on Disk X.
There are 3 files on Disk X.

fr_FR:
Il n'y a pas de fichiers sur Disk X.
Il y a un fichier sur Disk X.
Il y a 2 fichiers sur Disk X.
Il y a 3 fichiers sur Disk X.
```

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/i18n/format/choiceFormat.html
(Bentley, TutorialAtOracle Code Examples, 2013) InternationalisationDemo.java

Testing characters with Unicode properties.

In order to keep your code internationally adaptable never check for characters by using a limited ASCII range...

```
private static void demoTestingCharactersWithUnicodeProperties() {
  final String stringToSearch = "Lucy 78ϒ  利\u5229 Greek \u03B2 $.!()45";

  for (char charLoop : stringToSearch.toCharArray()) {
    // Don't do this, not unicode safe.
    if ((charLoop >= 'a' && charLoop <= 'z') || (charLoop >= 'A' && charLoop <= 'Z')) {
      System.out.printf("%c", charLoop);
    }
  }
}

// Output
LucyGreek  // Note legitmate non Ascii characters were wrongly filtered out.
```

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/i18n/text/charintro.html
(Bentley, TutorialAtOracle Code Examples, 2013) InternationalisationDemo.java

... Instead use unicode safe categories testing ...

Unicode safe category testing can be done using java.lang.Character methods.

```
//
private static void demoTestingCharactersWithUnicodeProperties() {
  final String stringToSearch = "Lucy 78ϒ  利\u5229 Greek \u03B2 $.!()45";

  for (char charLoop : stringToSearch.toCharArray()) {
    if (Character.isLetter(charLoop)) {
      System.out.printf("%c", charLoop);
    }
  }
}

// Output
Lucy利利Greekβ
```

(Oracle, 2012) http://docs.oracle.com/javase/tutorial/i18n/text/charintro.html
(Bentley, TutorialAtOracle Code Examples - HelloWorld, 2013) InternationalisationDemo.java

Common java.lang.Character testing methods:

- isDigit
- isLetter
- isLetterOrDigit
- isLowerCase
- isUpperCase
- isWhitespace
- isDefined

(Oracle, 2011) http://docs.oracle.com/javase/6/docs/api/java/lang/Character.html

Unicode safe category testing can be done using java.lang.Character.getType with Character Unicode constants.

```
final String stringToSearch = "Lucy 78ϒ 利\u5229 Greek \u03B2 $.!()45";

for (char charLoop : stringToSearch.toCharArray()) {
  switch (Character.getType(charLoop)) {
  case Character.UPPERCASE_LETTER:
    System.out.printf("%c: Character.UPPERCASE_LETTER%n", charLoop);
    break;
  case Character.LOWERCASE_LETTER:
    System.out.printf("%c: Character.LOWERCASE_LETTER%n", charLoop);
    break;
...

// Output
...
y: Character.LOWERCASE_LETTER
 : Character.SPACE_SEPARATOR
7: Character.DECIMAL_DIGIT_NUMBER
...
```

(Oracle, 2011) http://docs.oracle.com/javase/6/docs/api/java/lang/Character.html#getType%28char%29
(Bentley, TutorialAtOracle Code Examples - HelloWorld, 2013) InternationalisationDemo.java
http://developer.android.com/reference/java/lang/Character.html (for Unicode Category constants)

Unicode safe category testing can be done using Regex character sets using java.lang.Character or Unicode category constants (or other unicode sets of contants).

```
private static void demoTestingCharactersWithUnicodePropertiesRegexes() {
  final String stringToSearch = "Lucy 78ϒ  利\u5229 Greek \u03B2 $.!()45";

  // regex java.lang.Character character set
//    final String regex = "\\p{javaLetter}";

  // regex Unicode category constant character set
   final String regex = "\\p{L}";

  Pattern pattern = Pattern.compile(regex);
  Matcher matcher = pattern.matcher(stringToSearch);

  while (matcher.find()) {
    System.out.print(matcher.group());
  }
}

// Output
Lucy利利Greekβ
```

See (Bentley, Java Reference - Language - Regular Expressions.docx, 2013) &
http://developer.android.com/reference/java/lang/Character.html (for Unicode Category constants)
& http://docs.oracle.com/javase/6/docs/api/java/lang/Character.html (for java.lang.Character)

Comparing strings properly

Different locales may have different rules for comparision and sorting.
To compare strings properly, in a locale sensitive manner, don't use String.compareTo use Collator.compare.

```
// Default locale comparison
Collator myDefaultCollator = Collator.getInstance();
```

Example	Return Value	Explanation
myCollator.compare("abc", "def")	-1	"abc" is less than "def"
myCollator.compare("rtf", "rtf")	0	the two strings are equal
myCollator.compare("xyz", "abc")	1	"xyz" is greater than "abc"

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/i18n/text/locale.html

Demonstrate comparing and sorting strings properly.

```
private static class demoComparingStringsProperly {

  private static void sortStrings(java.text.Collator collator, String[] words) {
    String temp;

    // Use a more efficient sorting algorithm in practice.
    for (int i = 0; i < words.length; i++) {
      for (int j = i + 1; j < words.length; j++) {
        if (collator.compare(words[i], words[j]) > 0){
          temp = words[i];
          words[i] = words[j];
          words[j] = temp;
        }
      } // j
    } // i

  } // sortStrings

  private static void println1ArrayElements(Object[] array){
    for (Object element : array) {
```

```
      System.out.printf("  %s%n", element.toString());
    }
  } // printlnArrayElements

  private static void start() {
    String[] words = { "péché", "peach", "sin", "pêche" };
    Locale[] locales = { new Locale("en", "US"), new Locale("fr", "FR") };

    for (Locale localeLoop : locales) {
      Collator collator = Collator.getInstance(localeLoop);

      sortStrings(collator, words);
      System.out.println(localeLoop.toString() + ":");
      printlnArrayElements(words);
      System.out.println();
    }
  } // start

} // demoComparingStringsProperly

// Output. Note different order of words.
en_US:
  peach
  péché
  pêche
  sin

fr_FR:
  peach
  pêche
  péché
  sin
```

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/i18n/text/locale.html
(Bentley, TutorialAtOracle Code Examples - HelloWorld, 2013) InternationalisationDemo.java

You can define custom collation rules (sort order/comparison rules) by passing a custom string to the RuleBasedCollator.

```
String simpleRule = "< a < b < d < c";
RuleBasedCollator simpleCollator =  new RuleBasedCollator(simpleRule);
```

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/i18n/text/rule.html

If the sorting algorithm operates over a large number of entries you might want to optimise it. Optimise it by using a better sorting algorithm (e.g. mergesort or shellsort) and using a CollationKey class to operate on keys rather than strings.

Jlb: I don't see how using CollationKey would be faster given that you have to retrieve the actual string value for comparison anyway. E.g. "keys[i].compareTo(keys[j]) > 0"

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/i18n/text/perform.html

Unicode

See (Bentley, Java Reference - Language.docx, 2017) Strings > Unicode Representations.

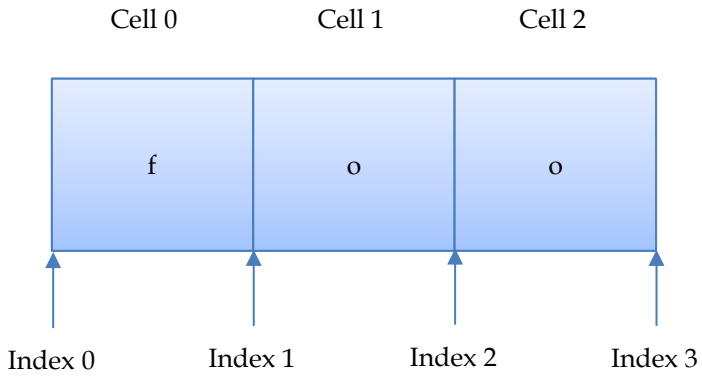
Detecting Text Boundaries

See also **Basic I/O – Streams**, especially **Scanning**.
The BreakIterator class provides locale-sensitive ways to detect four kinds of boundaries: character, word, sentence, and potential line break.

The BreakIterator controls a cursor, indexed from 0, with indices pointing to the cell boundaries, starting at the left edge, at zero. The end index comes **after** the cell of the last character. This last boundary equals the length of the string. This is as with regexes. You should use the BreakIterator class only with natural-language text. To tokenize a programming language, use the StreamTokenizer class (jlb: or perhaps the Scanner class). A user character may be composed of more than one Unicode character. see (Bentley, Java Reference - Language.docx, 2017), **Unicode > Overview**.
A **user character** may be composed of more than one Unicode character.

```
// the user character ü can be composed by combining the Unicode characters \u0075 (u) and
\u00a8 (¨). This isn't the best example, however, because the character ü may also be
represented by the single Unicode character \u00fc.


// In Arabic the word for house is:
ﻧُﻴْﺒَ

/// This word contains three user characters, but it is composed of the following six
Unicode characters:

String house = "\u0628" + "\u064e" + "\u064a" + "\u0652" + "\u067a" + "\u064f";
```

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/i18n/text/char.html



Character boundaries.

```
private static void demoBreakIterator() {
  String arabicForHouse = "\u0628\u064e\u064a\u0652\u067a\u064f";

  System.out.println(arabicForHouse);
  BreakIterator arCharIterator = BreakIterator.getCharacterInstance(new Locale("ar", "SA"));
  arCharIterator.setText(arabicForHouse);
  int boundary = arCharIterator.first();

  while (boundary != BreakIterator.DONE){
    System.out.printf("%d ", boundary);
    boundary = arCharIterator.next();
  }
}

// Output
ﻧُﻴْﺒَ
0 2 4 6
```

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/i18n/text/char.html
(Bentley, TutorialAtOracle Code Examples, 2013) InternationalisationDemo.java

Word boundaries. The word BreakIterator picks out words, even among punctuation in various languages.

```
private static void demoBreakIteratorWordBoundaryIlluminate() {
```

```
  String target = "She said, \"Hello There\". Cool.";
  StringBuilder markers = new StringBuilder();
  markers.setLength(target.length() + 1);
  BreakIterator iterator = BreakIterator.getWordInstance();

  for (int k = 0; k < markers.length(); k++) {
    markers.setCharAt(k, ' ');
  }

  iterator.setText(target);
  int boundary = iterator.first();

  while (boundary != BreakIterator.DONE) {
    markers.setCharAt(boundary, '^');
    boundary = iterator.next();
  }

  System.out.println(target);
  System.out.println(markers.toString());
}

// Output
She said, "Hello There". Cool.
^   ^^    ^^^^     ^^     ^^^^    ^^
```

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/i18n/text/word.html
(Bentley, TutorialAtOracle Code Examples, 2013) InternationalisationDemo.java

Word parse using the word BreakIterator.

```
private static void demoBreakIteratorWordParse(){
  String target = "She said, \"Hello There\". Cool.";
  BreakIterator iterator = BreakIterator.getWordInstance();

  iterator.setText(target);
  int start = iterator.first();
  int end = iterator.next();

  while (end != BreakIterator.DONE) {
    String word = target.substring(start, end);
    // Unicode safe check for a legitimate word (filter out punctuation and space).
    if (Character.isLetterOrDigit(word.codePointAt(0))) {
      System.out.println(word);
    }
    start = end;
    end = iterator.next();
  }
}

// Output
She
said
Hello
There
Cool
```

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/i18n/text/word.html
(Bentley, TutorialAtOracle Code Examples, 2013) InternationalisationDemo.java

Sentence boundaries.

```
private static void demoBreakIteratorSentence() {
  String target =
      "She stopped.  She said, \"Hello there,\" and then"
          + " went on. He's vanished!  What will we do?";
  BreakIterator iterator = BreakIterator.getSentenceInstance();

  iterator.setText(target);
  int start = iterator.first();
  int end = iterator.next();

  while (end != BreakIterator.DONE) {
```

```
    String sentence = target.substring(start, end);
    System.out.println(sentence);
    start = end;
    end = iterator.next();
  }
}

// Output
She stopped.
She said, "Hello there," and then went on.
He's vanished!
What will we do?
```

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/i18n/text/sentence.html
(Bentley, TutorialAtOracle Code Examples, 2013) InternationalisationDemo.java

Potential Line breaks. The BreakIterator is able to identify **potential** line breaks. It is up to the program to decide whether to use a potential line break to actually break the line.

```
private static void demoBreakIteratorPotentialLineBreak() {
  String target =
      "She stopped.  She said, \"Hello there,\" and then"
          + " There are twenty-four hours in a day.";
  int maxLineLength = 30;
  BreakIterator iterator = BreakIterator.getLineInstance();

  iterator.setText(target);
  int start = iterator.first();
  int end = iterator.next();
  int lineLength = 0;

  while (end != BreakIterator.DONE){
    String word = target.substring(start, end);
    lineLength = lineLength + word.length();
    if (lineLength >= maxLineLength){
      System.out.println();
      lineLength = word.length();
    }
    System.out.print(word);
    start = end;
    end = iterator.next();
  }
}

// Output
She stopped.  She said,
"Hello there," and then
There are twenty-four hours
in a day.
```

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/i18n/text/line.html
(Bentley, TutorialAtOracle Code Examples, 2013) InternationalisationDemo.java

Convert Latin digits to Non Latin digits

Convert latin digits ('0', '1', '2', etc) to non-Latin digits using three techniques:

- Specify a Locale variant and use NumberFormat. See **Setting the Locale** above.
- Use NumberShaper and specify the range (e.g. NumericShaper.ARABIC).
- Use NumberShaper with a getContextualShaper() instance.

Use NumberShaper and specify the range.

```
private static void convertLatinDigitsToNonLatinDigits() {
  String text = "Alphabetic text 0 1 2 3 4 5 6 7 8 9";
  char[] textChars = text.toCharArray();

  NumericShaper ns = NumericShaper.getShaper(NumericShaper.ARABIC);
```

```
  // shape() alters textChars
  ns.shape(textChars, 0, textChars.length);

  System.out.println(textChars);
}

// Output
Alphabetic text ٩ ٨ ٧ ٦ ٥ ٤ ٣ ٢ ١ ٠
```

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/i18n/text/shapedDigits.html
(Bentley, TutorialAtOracle Code Examples, 2013) WorkingWithText.java

Use NumberShaper with a getContextualShaper() instance. A contextual shaper changes the digits (if applicable) based on the language of the text that **precedes** the digits.

```
private static void convertLatinDigitsToNonLatinDigitsContextual() {
  // Arabic "precedes" because it is a right to left language (though digits are left to
  // right).
  String text =
      "123 (Not preceded by text), Cool 456 (English), 789.98 بـارد (Arabic), เย็น 123 (Thai)";
  char[] textChars = text.toCharArray();

  NumericShaper ns = NumericShaper.getContextualShaper(NumericShaper.ALL_RANGES);

  ns.shape(textChars, 0, textChars.length);

  System.out.println(textChars);
}

// Output
123 (Not preceded by text), Cool 456 (English), ٧٨٩.٩٨ بـارد (Arabic), เย็น ๑๒๓ (Thai)
```

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/i18n/text/shapedDigits.html
(Bentley, TutorialAtOracle Code Examples, 2013) WorkingWithText.java

Service Providers for Internationalisation

Service providers for internationalization enable the plug-in of locale-dependent data and services. You do this by implementing interfaces

java.util.spi	**java.text.spi**
CurrencyNameProvider LocaleServiceProvider TimeZoneNameProvider	BreakIteratorProvider CollatorProvider DateFormatProvider DateFormatSymbolsProvider DecimalFormatSymbolsProvider NumberFormatProvider

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/i18n/serviceproviders/index.html

Collections

Summary

A collection is an object that groups multiple elements.

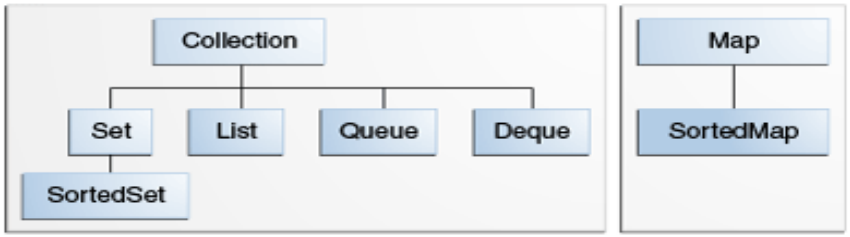
(Oracle, 2017) *http://docs.oracle.com/javase/tutorial/collections/intro/index.html*

The Java "collections framework" contains:

- Interfaces. You generally operate using these. Interfaces allow collections to be manipulated independently of their implementation. When you understand how to use these interfaces, you will know most of what there is to know about the Java Collections Framework.
- Implementations. Behind the scenes inner workings of the interfaces. As a rule, you should be thinking about the interfaces, not the implementations. For the most part, the choice of implementation affects only performance.
- Algorithms. Methods, common to many implementations/interfaces, that can act on the collection. E.g. Searching and sorting. These methods come from the Collections (note the plural) class.

(Oracle, 2017) *http://docs.oracle.com/javase/tutorial/collections/intro/index.html*
(Oracle, 2017) *http://docs.oracle.com/javase/tutorial/collections/implementations/index.html*

The core collection interfaces:



(Oracle, 2017) *http://docs.oracle.com/javase/tutorial/collections/interfaces/index.html*

A description of the core collection interfaces:

- **Collection**. The root of the collection hierarchy.
- **Set**. Forbids duplicates. E.g. Poker hand.
- **SortedSet**. A set (no duplicates) with elements sorted in ascending order. Uses natural ordering (e.g. alphabetical for strings); or Comparator order.
- **List**. Ordered collection. Can contain duplicates. You can control where the elements are inserted. You can reference elements with an numeric index.
- **Queue**. Typically orders elements by first-in, first out (FIFO). Useful for holding elements prior to processing. Priority queues, however, order by either according to the comparator or the elements natural ordering.
- **Deque**. A double ended queue. Supports insertion and removal at both ends. Can be used as a last-in-first-out (LIFO) stack or a first-in-first-out (FIFO) queue.
- **Map**. Stores key and value pairs. Cannot contain duplicate keys. The key maps to at most one value.
- **SortedMap.** A map with elements sorted in ascending key order. Uses natural ordering of key/value pairs (e.g. dictionary); or Comparator order.

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/collections/interfaces/index.html

Always perform coding operations against a Collection interface, not its implementation.

```
String[] folkArray = {"Xeno",
        "Marx",
        "Alphie",
        "Zara",
        "Marx",
        "Marx",
        "Hasselhoff",
        "Beiber"};
Set<String> set = new HashSet<String>();
for (String person : folkArray) {
    // We are operating against set, not HashSet.
    set.add(person);
}
System.out.printf("%d distinct words %s%n", set.size(), set);
```

This gives you the flexibility to change implementations with ease.

(Oracle, 2011) http://docs.oracle.com/javase/tutorial/collections/interfaces/set.html
C:\Users\John\Documents\Sda\Code\Java\Examples\TutorialAtOracle\HelloWorld\src\helloworld\demo\collections\SetDemo.java

Interfaces

Basics

All the core collection interfaces are defined with generics. When you declare an interface you can and should specify the type of object contained in the collection.

```
// The interface definitions
public interface Collection<E>...
public interface List<E> extends Collection<E>...

// Declare a collection with a type.
List<String> list = Arrays.asList("Hasselhoff", "Marx", "Beiber");
for (String s : list) {
    System.out.println(s);
}
```

```
// Output
Marx
Hasselhoff
Beiber
```

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/collections/interfaces/index.html

Declare collections in the following ways:

From scratch; then add. This allows you to grow the collection at a future point.

```
List<String> list = new ArrayList<String>();
list.add("George");
list.add("John");
list.add("Paul");
```

From Scratch; then Collections.addAll(). This allows you to grow the collection at a future point.

```
static private void creationAddAll() {
    List<String> list  = new ArrayList<String>();

    Collections.addAll(list, "Xeno", "Marx", "Alphie",
                       "Zara", "Marx", "Marx",
                       "Hasselhoff", "Beiber");
    System.out.println(list);
    list.add("Zoro");
    System.out.println(list);
}

// Output. List not immutable and can be grown.
[Xeno, Marx, Alphie, Zara, Marx, Marx, Hasselhoff, Beiber]
[Xeno, Marx, Alphie, Zara, Marx, Marx, Hasselhoff, Beiber, Zoro]
```

C:\Users\John\Documents\Sda\Code\Java\Examples\TutorialAtOracle\HelloWorld\src\helloworld\demo\collections\ListDemo.java
(Oracle, 2017) https://docs.oracle.com/javase/tutorial/collections/algorithms/index.html

Using the immutable multiple copy list ncopies().

```
static public void creation() {
    List<String> folk = new ArrayList<String>(Collections.nCopies(5, (String)null));
    System.out.println(folk);

    folk.add("Paul");
    System.out.println(folk);
    folk.add(0, "John");
    folk.add(2, "Ringo");
    System.out.println(folk);
}

// Output
[null, null, null, null, null]
[null, null, null, null, null, Paul]
[John, null, Ringo, null, null, null, null, Paul]
```

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/collections/implementations/convenience.html
C:\Users\John\Documents\Sda\Code\Java\Examples\TutorialAtOracle\HelloWorld\src\helloworld\demo\collections\ListDemo.java

From Arrays.asList (but this makes the collection of fixed length).

```
Collection<String> names = Arrays.asList("Xeno",
                                         "Marx",
                                         "Alphie",
                                         "Zara",
                                         "Hasselhoff"
```

```
                                        "Beiber");

List<String> list = Arrays.asList("Marx", "Hasselhoff", "Beiber");
list.add("Bill"); // Throws java.lang.UnsupportedOperationException


private static List<String> mList = new ArrayList<String>(Arrays.asList("Xeno",
                                                                         "Marx",
                                                                         "Alphie",
                                                                         "Zara",
                                                                         "Marx",
                                                                         "Marx",
                                                                         "Hasselhoff",
                                                                         "Beiber"));

Set<String> set = new HashSet<String>(Arrays.asList("Xeno",
                                                    "Marx",
                                                    "Alphie",
                                                    "Zara",
                                                    "Hasselhoff",
                                                    "Beiber"));

SortedSet<String> sortedSet = new TreeSet<>(Arrays.asList("Xeno",
                                                          "Marx",
                                                          "Alphie",
                                                          "Zara",
                                                          "Hasselhoff",
                                                          "Beiber"));
```

C:\Users\John\Documents\Sda\Code\Java\Examples\TutorialAtOracle\HelloWorld\src\helloworld\demo\CollectionsDemo.javas

Operate on a collection of custom objects.

```
static private void listOfCustomObjectDemo() {
    List<Bicycle> bikes = new ArrayList<Bicycle>();
    bikes.add(new Bicycle("Malven", "Star", 3, 34));
    bikes.add(new Bicycle("Acme", "BMS", 5, 11));
    System.out.println(bikes);
}

// Output
[(Malven Star) cadence: 3 speed: 34 number of gears: 12
, (Acme BMS) cadence: 5 speed: 11 number of gears: 12
]
```

C:\Users\John\Documents\Sda\Code\Java\Examples\TutorialAtOracle\HelloWorld\src\helloworld\demo\collections\ListDemo.java
(Bentley, Java Reference - Language.docx, 2017) "Bicycle".

Collection Interface

Conversion constructor

All core collection implementations have a constructor that takes a collection argument. This constructor is known as the "conversion constructor". It allows you to convert one collection to another.

```
// Start with a set collection
Set<String> set = new HashSet<String>(Arrays.asList("Xeno",
                                                    "Marx",
                                                    "Alphie",
                                                    "Zara",
                                                    "Hasselhoff",
                                                    "Beiber"));

System.out.println(set);
```

```
// Use the conversion constructor to convert to a SortedSet collection.
SortedSet<String> sortedSet = new TreeSet<>(set);
System.out.println(sortedSet);

// Output
[Zara, Alphie, Xeno, Beiber, Marx, Hasselhoff]
[Alphie, Beiber, Hasselhoff, Marx, Xeno, Zara]
```

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/collections/interfaces/collection.html
C:\Users\John\Documents\Sda\Code\Java\Examples\TutorialAtOracle\HelloWorld\src\helloworld\demo\CollectionsDemo.java

Collection methods

A collection interface has methods that perform basic operations. These are inherited by the specialized collections ...

Element Methods:

```
contains()
add()
remove()
iterator()
```

Entire collection Methods:

```
size()
isEmpty()
containsAll()
addAll()
removeAll()
retainAll()
clear()
SortedSet<String> sortedSet = new TreeSet<>(Arrays.asList("Xeno",
                                                          "Marx",
                                                          "Alphie",
                                                          "Zara",
                                                          "Hasselhoff",
                                                          "Beiber"));

sortedSet.add("Harris");
System.out.println(sortedSet);
System.out.println("size: " + sortedSet.size());

// Output
[Alphie, Beiber, Harris, Hasselhoff, Marx, Xeno, Zara]
size: 7
```

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/collections/interfaces/collection.html

Collection Iteration

Iterate a collection in four ways:

1. aggregate operations (from Java 8);
2. Enhanced-for, elements referenced by object;

```
for (Object obj : sortedSet) {
    System.out.println(obj.toString());
}
```

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/collections/interfaces/collection.html

3. Enhanced-for, elements referenced by type;

```
for (String s : sortedSet) {
    System.out.println(s);
}
```

(Oracle, 2012) http://docs.oracle.com/javase/tutorial/collections/interfaces/collection.html

4. Enhanced-for, iterators.

General use of an iterator.

```
static void filter(Collection<?> c) {
    for (Iterator<?> it = c.iterator(); it.hasNext(); )
        if (!cond(it.next()))
            it.remove();
}
```

(Oracle, 2012) http://docs.oracle.com/javase/tutorial/collections/interfaces/collection.html

Iterators allow you to taverse a collection and selectively remove elements from the collection. remove(), removes the last element accessed by hasNext(). Iterator.remove is the only safe way to modify a collection during iteration. Use an iterator instead of a for-each when you want to either: remove elements; or iterate over multiple collections in parallel.

```
// The Interator Interface
public interface Iterator<E> {
    boolean hasNext();
    E next();
    void remove(); //optional
}

// Example removal
SortedSet<String> sortedSet = new TreeSet<>(Arrays.asList("Xeno",
                                                          "Marx",
                                                          "Alphie",
                                                          "Zara",
                                                          "Hasselhoff",
                                                          "Beiber"));

System.out.println(sortedSet);
for (Iterator<String> iterator = sortedSet.iterator(); iterator.hasNext();) {
    if (iterator.next() == "Marx") {
        iterator.remove();
    }
}
System.out.println(sortedSet);

// Output
[Alphie, Beiber, Hasselhoff, Marx, Xeno, Zara]
[Alphie, Beiber, Hasselhoff, Xeno, Zara]
```

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/collections/interfaces/collection.html
C:\Users\John\Documents\Sda\Code\Java\Examples\TutorialAtOracle\HelloWorld\src\helloworld\demo\CollectionsDemo.java

See also:

- List Iterator
- EnumSet Range
- Map Iteration
- EnumMap

Array Translations

To get a collection from an array use Arrays.asList(). But this will mean the size of the List cannot be changed.

```
List<String> list = Arrays.asList("Marx", "Hasselhoff", "Beiber");

Set<String> set = new HashSet<String>(Arrays.asList("Xeno",
                                                    "Marx",
                                                    "Alphie",
                                                    "Zara",
                                                    "Hasselhoff",
                                                    "Beiber"));

SortedSet<String> sortedSet = new TreeSet<>(Arrays.asList("Xeno",
                                                          "Marx",
                                                          "Alphie",
                                                          "Zara",
                                                          "Hasselhoff",
                                                          "Beiber"));
```

C:\Users\John\Documents\Sda\Code\Java\Examples\TutorialAtOracle\HelloWorld\src\helloworld\demo\CollectionsDemo.java
(Oracle, 2017) http://docs.oracle.com/javase/tutorial/collections/implementations/convenience.html

To get an array from a collection use toArray().

```
// Convert collection to array of general objects
Object[] arrayOfObjects = collection.toArray();
for (Object item : arrayOfObjects) {
    System.out.println(item.toString());
}

System.out.println();

// Convert collection to array of typed objects
// The following dumps the collection into array whose length is determined
// By the number of elements in the collection.
String[] arrayOfStrings = collection.toArray(new String[0]);
for (String item : arrayOfStrings) {
    System.out.println(item);
}
```

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/collections/interfaces/collection.html
C:\Users\John\Documents\Sda\Code\Java\Examples\TutorialAtOracle\HelloWorld\src\helloworld\demo\CollectionsDemo.java

Set Interface

Basics

A Set is a collection that forbids duplicates. E.g. Poker hand.

```
static private void basics() {
    String[] folkArray = {"Xeno",
            "Marx",
            "Alphie",
            "Zara",
```

```
            "Marx",
            "Marx",
            "Hasselhoff",
            "Beiber"};
    Set<String> set = new HashSet<St
    for (String person : folkArray) {
        // We are operating against set, n
        set.add(person);
    }
    System.out.println(set);
}

// Output
[Zara, Alphie, Xeno, Beiber, Marx, Hasselhoff]
```

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/collections/interfaces/set.html

A Set interface only contains the methods inherited from Collection.

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/collections/interfaces/set.html

For a Set choose an implementation thus:

- HashSet. Most common. Best performing. No ordering.
- TreeSet. Orders elements based on their value. Substantially slower than HashSet.
- LinkedHashSet. Orders elements based on the order that elements were inserted into the set ("insertion-order"). Slightly slower than HashSet.

(Oracle, 2012) http://docs.oracle.com/javase/tutorial/collections/interfaces/set.html

You don't have to create a SortedSet in order to sort a set. You can just use the TreeSet or LinkedHashSet implementations on a Set, for the effects noted above:

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/collections/interfaces/set.html

Remove Duplicates

Return a collection with duplicates removed. Use ...

Collection<Type> noDups = new HashSet<Type>(c);

... which creates a Set (thereby having no duplicates) by leveraging the Conversion constructor.

```
List<String> list = Arrays.asList("Xeno",
                                  "Marx",
                                  "Alphie",
                                  "Zara",
                                  "Marx",
                                  "Marx",
                                  "Hasselhoff",
                                  "Beiber");

System.out.println(list);
Collection<String> noDuplicatesCollection = new HashSet<String>(list);
System.out.println(noDuplicatesCollection);

// Output
[Xeno, Marx, Alphie, Zara, Marx, Marx, Hasselhoff, Beiber]
[Zara, Alphie, Xeno, Beiber, Marx, Hasselhoff]
```

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/collections/interfaces/set.html

Return a collection with duplicates removed, using a a LinkedHashSet:

Collection<Type> noDups = new LinkedHashSet<Type>(c);

(Oracle, 2012) http://docs.oracle.com/javase/tutorial/collections/interfaces/set.html

Remove duplicates using a method:

```
// Remove duplicates method
public static <E> Set<E> removeDuplicates(Collection<E> c) {
    return new LinkedHashSet<E>(c);
}

// Code
List<String> list = Arrays.asList("Xeno",
                                  "Marx",
                                  "Alphie",
                                  "Zara",
                                  "Marx",
                                  "Marx",
                                  "Hasselhoff",
                                  "Beiber");
System.out.println(list);
Collection<String> noDuplicatesCollection = removeDuplicates(list);
System.out.println(noDuplicatesCollection);

// Output
[Xeno, Marx, Alphie, Zara, Marx, Marx, Hasselhoff, Beiber]
[Xeno, Marx, Alphie, Zara, Hasselhoff, Beiber]
```

(Oracle, 2012) http://docs.oracle.com/javase/tutorial/collections/interfaces/set.html

Ensure elements are unique with set:

```
String[] folkArray = {"Xeno",
        "Marx",
        "Alphie",
        "Zara",
        "Marx",
        "Marx",
        "Hasselhoff",
        "Beiber"};
Set<String> set = new HashSet<String>();
for (String person : folkArray) {
    // We are operating against set, not HashSet.
    set.add(person);
}
System.out.printf("%d distinct words %s%n", set.size(), set);

// Output
6 distinct words [Zara, Alphie, Xeno, Beiber, Marx, Hasselhoff]
```

(Oracle, 2011) http://docs.oracle.com/javase/tutorial/collections/interfaces/set.html
C:\Users\John\Documents\Sda\Code\Java\Examples\TutorialAtOracle\HelloWorld\src\helloworld\demo\collections\SetDemo.java

List Interface

Basics

A List interface is an ordered collection (sometimes called a "sequence"). Lists may contain duplicates. By default (always?), Lists maintain insertion order.

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/collections/interfaces/list.html

List Methods

General

In addition to the collection methods List offers the following.

Category	Method	Description
Positional access	get(int) set(int, E) add(int, E) addAll(int, Collection<? Extends E>) remove(int)	Manipulates elements based on their numerical position in the list.
Search	indexOf(Object) lastIndexOf(Object)	Search for a specified object in the list and return its numerical position
Iteration	listIterator() listIterator(int)	Extends Iterator semantics to take advantage of the list's sequential nature.
Range-view	subList(int, int)	Perform arbitrary range operations on the list.

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/collections/interfaces/list.html

Positional access example using swap().

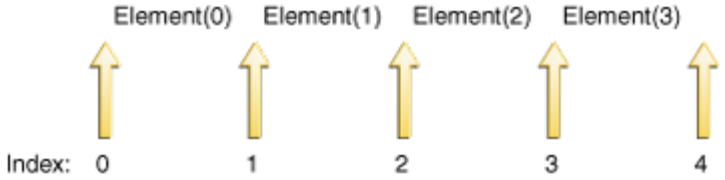
```
public class ListDemo {
    private static List<String> mList = Arrays.asList("Xeno",
                                                       "Marx",
                                                       "Alphie",
                                                       "Zara",
                                                       "Marx",
                                                       "Marx",
                                                       "Hasselhoff",
                                                       "Beiber");

    static public void start() {
        positionalAccess();
    }

    static private void positionalAccess() {
        System.out.println(mList);
        swap(mList, 2, 6);
        System.out.println(mList);
    }

    // swap is implemented in the Collections class.
    public static <E> void swap(List<E> list, int i, int j) {
        E temp = list.get(i);
        list.set(i, list.get(j));
        list.set(j, temp);
    }
}


// Output
[Xeno, Marx, Alphie, Zara, Marx, Marx, Hasselhoff, Beiber]
[Xeno, Marx, Hasselhoff, Zara, Marx, Marx, Alphie, Beiber]
```

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/collections/interfaces/list.html
C:\Users\John\Documents\Sda\Code\Java\Examples\TutorialAtOracle\HelloWorld\src\helloworld\demo\collections\ListDemo.java

List Iterator

List Iterator. A List inherits iterator() from the collection class. But List also have a ListIterator which provides additional benefits: it can more backwards; the list can be modified during iterations, and the current position of the iterator can be obtained.

A list index points to a position before the cursor. So in a list of length n (with n elements), there are n+1 valid values for index, from 0 to n, inclusive. So the index of a cursor is always between two elements.



(Oracle, 2017) http://docs.oracle.com/javase/tutorial/collections/interfaces/list.html

When obtaining the index during a ListIteration use iterator.nextIndex() if traversing backwards, and iterator.previousIndex() if traversing forwards.

```
System.out.println(mList);
for (ListIterator<String> it = mList.listIterator(mList.size()); it.hasPrevious();) {
    String item = it.previous();
    System.out.printf("%d %s%n", it.nextIndex(), item );
}

// Output
[Xeno, Marx, Alphie, Zara, Marx, Marx, Hasselhoff, Beiber]
7 Beiber
6 Hasselhoff
5 Marx
4 Marx
3 Zara
2 Alphie
1 Marx
0 Xeno
```

To manipulate the List entries use remove(), set(), or add(). Remove() works on the last element returned by next. set()works on the last element returned by next() or previous(). Add() inserts a new element before the current cursor position.

```
// In practice we'd use Collections.replaceAll
static private <E> void replace(List<E> list, E val, E newValue) {
    for (ListIterator<E> it = list.listIterator(); it.hasNext(); ) {
        if (val == null ? it.next() == null : val.equals(it.next())) {
            it.set(newValue);
        }
    }
}
```

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/collections/interfaces/list.html
C:\Users\John\Documents\Sda\Code\Java\Examples\TutorialAtOracle\HelloWorld\src\helloworld\demo\collections\ListDemo.java

Range-View Operations

A range-view is a sublist of an original list.

```
// Get a range-view
System.out.println(mList);
```

```
List<String> subList = mList.subList(2, 5);
System.out.println(subList);

// Output
[Xeno, Marx, Alphie, Zara, Marx, Marx, Hasselhoff, Beiber]
[Alphie, Zara, Marx]
```

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/collections/interfaces/list.html
C:\Users\John\Documents\Sda\Code\Java\Examples\TutorialAtOracle\HelloWorld\src\helloworld\demo\collections\ListDemo.java

Because a range-view it is a "view" changes in the range-view effect the original list and vica versa.

```
// Modifying one list effects the other.
subList.set(0, "Fonzi");
System.out.println(subList);
System.out.println(mList);

// Output
[Fonzi, Zara, Marx]
[Xeno, Marx, Fonzi, Zara, Marx, Marx, Hasselhoff, Beiber]
```

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/collections/interfaces/list.html
C:\Users\John\Documents\Sda\Code\Java\Examples\TutorialAtOracle\HelloWorld\src\helloworld\demo\collections\ListDemo.java

List Algorithms

Basics

Most of the "Collections algorithms", methods in the Collection class, are for operations on List instances. See also Algorithms.

List Algorithms (from the Collections class) ...

Name	Description
sort	sorts a List using a merge sort algorithm, which provides a fast, stable sort. (A stable sort is one that does not reorder equal elements.)
shuffle	randomly permutes the elements in a List.
reverse	reverses the order of the elements in a List.
rotate	rotates all the elements in a List by a specified distance.
swap	swaps the elements at specified positions in a List.
replaceAll	replaces all occurrences of one specified value with another.
fill	overwrites every element in a List with the specified value.
copy	copies the source List into the destination List.
binarySearch	searches for an element in an ordered List using the binary search algorithm.
indexOfSubList	returns the index of the first sublist of one List that is equal to another.
lastIndexOfSubList	returns the index of the last sublist of one List that is equal to another.

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/collections/interfaces/list.html

Detail

Swap.

```
private static List<String> mList = Arrays.asList("Xeno",
                                                   "Marx",
                                                   "Alphie",
                                                   "Zara",
                                                   "Marx",
                                                   "Marx",
                                                   "Hasselhoff",
                                                   "Beiber");

static private void swapDemo() {
    System.out.println(mList);
    Collections.swap(mList, 2, 6);
    System.out.println(mList);
}

// Output
[Xeno, Marx, Alphie, Zara, Marx, Marx, Hasselhoff, Beiber]
[Xeno, Marx, Hasselhoff, Zara, Marx, Marx, Alphie, Beiber]
```

C:\Users\John\Documents\Sda\Code\Java\Examples\TutorialAtOracle\HelloWorld\src\helloworld\demo\collections\ListDemo.java

Shuffle. Random order on every run.

```
private static List<String> mList = Arrays.asList("Xeno",
                                                   "Marx",
                                                   "Alphie",
                                                   "Zara",
                                                   "Marx",
                                                   "Marx",
                                                   "Hasselhoff",
                                                   "Beiber");

static private  void shuffleDemo() {
    System.out.println(mList);
    Collections.shuffle(mList);
    System.out.println(mList);
}

// Output
[Xeno, Marx, Alphie, Zara, Marx, Marx, Hasselhoff, Beiber]
[Beiber, Marx, Hasselhoff, Zara, Marx, Alphie, Xeno, Marx]
```

C:\Users\John\Documents\Sda\Code\Java\Examples\TutorialAtOracle\HelloWorld\src\helloworld\demo\collections\ListDemo.java

Queue Interface

Typically orders elements by first-in, first out (FIFO). Useful for holding elements prior to processing. Priority queues, however, order by either according to the comparator or the elements natural ordering.

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/collections/interfaces/queue.html

In addition the basic collection operations queues provide additional insertion, removal, and inspection operations.

The Queue interface:

```
public interface Queue<E> extends Collection<E> {
    E element();
```

```
    boolean offer(E e);
    E peek();
    E poll();
    E remove();
}
```

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/collections/interfaces/queue.html

Queue methods are divided according to what happens if an operation fails: thrown an exception; or return a code (null or false, depending on the operation).

Queue Interface Structure		
Type of Operation	**Throws exception**	**Returns special value**
Insert	add(e)	offer(e)
Remove	remove()	poll()
Examine	element()	peek()

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/collections/interfaces/queue.html

Whatever the ordering of a queue the "head" of the queue is the element that would be removed by a call to remove() or poll().

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/collections/interfaces/queue.html

Every `Queue` implementation must specify its ordering properties.

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/collections/interfaces/queue.html

It is possible for a Queue implementation to restrict the number of elements that it holds; such queues are known as bounded. Some Queue implementations in java.util.concurrent are bounded, but the implementations in java.util are not.

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/collections/interfaces/queue.html

Do not insert nulls into a queue (which you can do in LinkedList implementations).

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/collections/interfaces/queue.html

Queue example (a contrived example, you wouldn't do this in practice).

```
static private void countDown() throws InterruptedException {
    int time = 10;
    Queue<Integer> queue = new LinkedList<Integer>();

    for (int i = 0; i < time; i++) {
        queue.add(i);
    }

    while (!queue.isEmpty()) {
        System.out.println(queue.remove());
        Thread.sleep(1000);
    }
}

// Output
0
1
2
```

```
3
...
```

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/collections/interfaces/queue.html

Queue example with a PriorityQueue implementation (a contrived example, you wouldn't do this in practice).

```
static private <E> List<E> heapSort(Collection<E> c) {
    // By default a PriorityQueue use natural element ordering. In the case of strings,
    // ordering alphabetically.
    Queue<E> queue = new PriorityQueue<E>(c);
    System.out.printf("%s PriorityQueue%n", queue);
    List<E> resultList = new ArrayList<E>();

    while (!queue.isEmpty()) {
        resultList.add(queue.remove());
    }

    return resultList;
}

private static List<String> mList = Arrays.asList("Xeno",
                                                  "Marx",
                                                  "Alphie",
                                                  "Zara",
                                                  "Marx",
                                                  "Marx",
                                                  "Hasselhoff",
                                                  "Beiber");

static private void priorityQueueDemo() {
    List<String> resultList = null;
    System.out.println(mList);
    resultList = heapSort(mList);
    System.out.println(resultList);
}

// Output
[Xeno, Marx, Alphie, Zara, Marx, Marx, Hasselhoff, Beiber]
[Alphie, Beiber, Hasselhoff, Marx, Marx, Marx, Xeno, Zara] PriorityQueue
[Alphie, Beiber, Hasselhoff, Marx, Marx, Marx, Xeno, Zara]
```

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/collections/interfaces/queue.html
C:\Users\John\Documents\Sda\Code\Java\Examples\TutorialAtOracle\HelloWorld\src\helloworld\demo\collections\QueueDemo.java

Deque Interface

A deque (pronounced "deck") is a double ended queue. Supports insertion and removal at both ends. Can be used as a last-in-first-out (LIFO) stack or a first-in-first-out (FIFO) queue.

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/collections/interfaces/deque.html

Deque interface methods.

Type of Operation	First Element (Beginning of the Deque instance)	Last Element (End of the Deque instance)
Insert	addFirst(e) offerFirst(e) throws exception when deque restricted	addLast(e) offerLast(e) throws exception when deque restricted
Remove	removeFirst() throws exception when deque empty pollFirst() returns null when deque empty	removeLast()throws exception when deque empty pollLast()returns null when deque empty

	removeFirstOccurence() remove specified element if exist. Return true if element exists.	removeLastOccurence()remove specified element if exist. Return true if element exists.
Examine	getFirst() throws exception when deque empty peekFirst() returns null when deque empty	getLast()throws exception when deque empty peekLast()returns null when deque empty

(Oracle, 2017) *http://docs.oracle.com/javase/tutorial/collections/interfaces/deque.html*

Map Interface

Map Basics

A map stores key and value pairs. Cannot contain duplicate keys. The key maps to at most one value.

(Oracle, 2017) *http://docs.oracle.com/javase/tutorial/collections/interfaces/map.html*

Map interface methods:

- Basic Operations: put, get, remove, containsKey, containsValue, size, and isEmpty.
- Bulk operations: putAll and clear.
- Views: keySet, entrySet, and values.

(Oracle, 2017) *http://docs.oracle.com/javase/tutorial/collections/interfaces/map.html*

But you can use collection methods on the set or collection returned by a collection view. (See below).

Map basic example.

```
static private void frequencyTable(){
    String[] keys = {"Xeno",
            "Marx",
            "Alphie",
            "Zara",
            "Marx",
            "Marx",
            "Hasselhoff",
            "Beiber"};
    Map<String, Integer> map = new HashMap<String, Integer>();

    Integer value = null;
    for (String key : keys) {
        // Check if key already in map
        Integer priorFrequency = map.get(key);
        Integer frequency = (priorFrequency == null) ? 1 : priorFrequency + 1;
        map.put(key, frequency);
    }

    System.out.printf("%d distinct words: %n", map.size());
    System.out.println(map);
}

// Output
6 distinct words:
{Zara=1, Alphie=1, Xeno=1, Beiber=1, Marx=3, Hasselhoff=1}
```

(Oracle, 2017) *http://docs.oracle.com/javase/tutorial/collections/interfaces/map.html*
C:\Users\John\Documents\Sda\Code\Java\Examples\TutorialAtOracle\HelloWorld\src\helloworld\demo\collections\MapDemo.java

To set a value given a key, use map.put(key, value).

```
Map<String, Integer> map = getFrequencyTableMap();
System.out.println(map);

// Change value given a key
map.put("Beiber", 10);
System.out.println(map);

// Output
{Zara=1, Alphie=1, Xeno=1, Beiber=1, Marx=3, Hasselhoff=1}
{Zara=1, Alphie=1, Xeno=1, Beiber=10, Marx=3, Hasselhoff=1}
```

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/collections/interfaces/map.html
C:\Users\John\Documents\Sda\Code\Java\Examples\TutorialAtOracle\HelloWorld\src\helloworld\demo\collections\MapDemo.java

To output the above map in alphabetical order you only need to change the implementation to TreeMap, without having to change the interface to SortedMap.

```
static private void frequencyTable(){
    ...
    Map<String, Integer> map = new TreeMap<String, Integer>();
    ...
}

// Output
6 distinct words:
{Alphie=1, Beiber=1, Hasselhoff=1, Marx=3, Xeno=1, Zara=1}
```

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/collections/interfaces/map.html
C:\Users\John\Documents\Sda\Code\Java\Examples\TutorialAtOracle\HelloWorld\src\helloworld\demo\collections\MapDemo.java

Likewise to preserve insertion order, just change the implementation to LinkedHashMap.

```
static private void frequencyTable(){
    ...
    Map<String, Integer> map = new LinkedHashMap<String, Integer>();
    ...
}

// Output
6 distinct words:
{Xeno=1, Marx=3, Alphie=1, Zara=1, Hasselhoff=1, Beiber=1}
```

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/collections/interfaces/map.html
C:\Users\John\Documents\Sda\Code\Java\Examples\TutorialAtOracle\HelloWorld\src\helloworld\demo\collections\MapDemo.java

Two Map instances are equal if they represent the same key-value mappings.

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/collections/interfaces/map.html

Map conversion constructor. It converts one map to another, regardless of the implementation type of either map.

```
Map<K, V> copy = new HashMap<K, V>(m);
```

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/collections/interfaces/map.html

Map Bulk Operations

One subtle use of the putAll bulk operation with the map conversion constructor allows map initialisation that overrides defaults.

```
static private <K, V> Map<K, V> getNewMapWithDefaults(Map<K, V> defaults, Map<K,
        V> overrides) {
    Map<K, V> result = new LinkedHashMap(defaults);
    result.putAll(overrides);
    return result;
}

static private void newMapOverrideDefaultsDemo(){
    Map<String, String> defaults = new LinkedHashMap<String, String>();
    defaults.put("Sydney", "FTTP");
    defaults.put("Melbourne", "FTTP");
    defaults.put("Darwin", "FTTP");
    defaults.put("Brisbane", "FTTP");

    Map<String, String> overrides = new HashMap<String, String>();
    defaults.put("Melbourne", "Copper");
    defaults.put("Darwin", "HFC");

    Map<String, String> currentPlan = getNewMapWithDefaults(defaults, overrides);
    System.out.println(currentPlan);
}

// Output
{Sydney=FTTP, Melbourne=Copper, Darwin=HFC, Brisbane=FTTP}
```

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/collections/interfaces/map.html
C:\Users\John\Documents\Sda\Code\Java\Examples\TutorialAtOracle\HelloWorld\src\helloworld\demo\collections\MapDemo.java

Collection Views

A "collection view" of a map is a representation of a map, or part of a map, as a collection (including the Set descendant). There are three collection views of a map:

- keySet, map.keySet(). A **Set** of keys of the map.
- values, map.values(). A **Collection** of values of the map. It is not a set because multiple there can be multiple entries of the same value.
- entrySet, map.entrySet(). A **Set** of key-value paris contained in the Map. The Map interface provides a nested interface Map.Entry<K, V>.

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/collections/interfaces/map.html

The map interface does not inherit from the collection interface.

Check this for yourself in ItelliJ IDEA.

Map Iteration

A collection view provides the only means to iterate over a map. You can use any of the three collections views to perform an interation.

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/collections/interfaces/map.html

You can interate over a map, with a collection view, by using a for-each construct or an iterator.

- Start with a map for our example.

```
Map<String, Integer> map = getFrequencyTableMap();
System.out.println(map);

// {Zara=1, Alphie=1, Xeno=1, Beiber=1, Marx=3, Hasselhoff=1}
```

- Key iteration.

```
// Iterate over keys, using for-each
for (String key : map.keySet()) {
    System.out.println(key);
}

// Iterate over keys, using iterator
for (Iterator<String> it = map.keySet().iterator(); it.hasNext(); ) {
    if (it.next() == "Marx")
        it.remove();
// {Zara=1, Alphie=1, Xeno=1, Beiber=1, Hasselhoff=1}
```

- Values iteration.

```
// Iterate over values, using for-each
for (Integer value : map.values()) {
    System.out.printf("%d ", value);
}
// 1 1 1 1 3 1

// Iterate over values, using iterator
for (Iterator<Integer> it = map.values().iterator(); it.hasNext(); ) {
    if (it.next() == 3) it.remove();
// {Zara=1, Alphie=1, Xeno=1, Beiber=1, Hasselhoff=1}
```

- Entry Set iteration.

```
// Iterate over entry set, using for-each
for (Map.Entry<String, Integer> e : map.entrySet()) {
    System.out.println(e.getKey() + ": " + e.getValue());
}

// Iterate over entry set, using iterator
for (Iterator it = map.entrySet().iterator(); it.hasNext();) {
    Map.Entry<String, Integer> entry = (Map.Entry) it.next();
    if (entry.getKey() == "Beiber")
        it.remove();
}
```

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/collections/interfaces/map.html

With all three Collection views, calling an Iterator's remove operation removes the associated entry from the backing Map, assuming that the backing Map supports element removal to begin with.

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/collections/interfaces/map.html

To change a value during an iteration use a Map.Entry's setValue(). This assumes the Map support value modification. This is the only safe way to do it (during iteration).

```
Map<String, Integer> map = getFrequencyTableMap();
System.out.println(map);

// Change value during iteration
for (Map.Entry<String, Integer> e : map.entrySet()) {
    if (e.getKey() == "Xeno")
        e.setValue(12);
}
```

```
System.out.println(map);

// Output
{Zara=1, Alphie=1, Xeno=1, Beiber=1, Marx=3, Hasselhoff=1}
{Zara=1, Alphie=1, Xeno=12, Beiber=1, Marx=3, Hasselhoff=1}
```

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/collections/interfaces/map.html
C:\Users\John\Documents\Sda\Code\Java\Examples\TutorialAtOracle\HelloWorld\src\helloworld\demo\collections\MapDemo.java

Collection views support removal in its many forms: remove, removeAll, retainAll, and clear operations, as well as the Iterator.remove operation (assuming the map supports element removal).

(Oracle, 2012) http://docs.oracle.com/javase/tutorial/collections/interfaces/map.html

Collection views never support addition. To perform addition use the Map's put or putAll methods.

(Oracle, 2012) http://docs.oracle.com/javase/tutorial/collections/interfaces/map.html

Using Collection Views for Map Algebra

You can use Collection bulk operations, containsAll, removeAll, and retainAll on a Map's collection views, and Map methods, to perform Map Algebra. The rest of this section goes through some of these Map Algebra possibilities.

(Oracle, 2012) http://docs.oracle.com/javase/tutorial/collections/interfaces/map.html

Informational

Get map that contains all the key-value mappings in the second.

```
if (m1.entrySet().containsAll(m2.entrySet())) {
    ...
}
```

(Oracle, 2012) http://docs.oracle.com/javase/tutorial/collections/interfaces/map.html

Get whether two map objects contain mappings for all of the same keys.

```
if (m1.keySet().equals(m2.keySet())) {
    ...
}
```

(Oracle, 2012) http://docs.oracle.com/javase/tutorial/collections/interfaces/map.html

Get keys common to two maps (the code is similar for getting values common to two maps).

```
Set<KeyType>commonKeys = new HashSet<KeyType>(m1.keySet());
commonKeys.retainAll(m2.keySet());
```

(Oracle, 2012) http://docs.oracle.com/javase/tutorial/collections/interfaces/map.html

Destructive

Remove all the key-value pairs that one map has in common with another.

```
m1.entrySet().removeAll(m2.entrySet());
```

(Oracle, 2012) *http://docs.oracle.com/javase/tutorial/collections/interfaces/map.html*

Remove from one map all of the keys that have mappings in another

```
m1.keySet().removeAll(m2.keySet());
```

Multimaps

A multimap is a map with a key to multiple values. Java doesn't include a multimap interface as such, but you can use a regular map whose values are list instances.

```
Map<String, List<String>> m = new HashMap<String, List<String>>();
```

See anagram example in (Oracle, 2017) *http://docs.oracle.com/javase/tutorial/collections/interfaces/map.html*

Sorted Set Interface

Intro

A SortedSet is a set (no duplicates) with elements sorted in ascending order. Uses natural ordering (e.g. alphabetical for strings); or Comparator order. It has only one implementation, the TreeSet.

```
private static SortedSet<String> mSortedSet = new TreeSet<String>(
        Arrays.asList("Xeno", "Marx", "Alphie", "Zara", "Marx", "Marx", "Hasselhoff",
                      "Beiber"));
static public void start() {
    System.out.println(mSortedSet);
}

// Output: No duplicates and ordered (naturally).
[Alphie, Beiber, Hasselhoff, Marx, Xeno, Zara]
```

(Oracle, 2017) *http://docs.oracle.com/javase/tutorial/collections/interfaces/sorted-set.html*

SortedSet interface:

```
public interface SortedSet<E> extends Set<E> {
    // Range-view
    SortedSet<E> subSet(E fromElement, E toElement);
    SortedSet<E> headSet(E toElement);
    SortedSet<E> tailSet(E fromElement);

    // Endpoints
    E first();
    E last();

    // Comparator access
    Comparator<? super E> comparator();
}
```

(Oracle, 2017) *http://docs.oracle.com/javase/tutorial/collections/interfaces/sorted-set.html*

Most SortedSet operations inherit from Set.

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/collections/interfaces/sorted-set.html

When using the SortedSet conversion constructor (through the TreeSet implementation, the only implementation) you can use several different constructors for various purposes:

- TreeSet(java.util.Collection<? extends E>). Returns a SortedSet, sorted according to natural ordering.
- TreeSet(java.util.Comparator<? super E>). Returns an empty SortedSet, sorted according to the comparator.
- TreeSet(java.util.SortedSet<E>). Returns another SortedSet instance, with the same elements, sorted according to the passed SortedSet's ordering. [What for??]

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/collections/interfaces/sorted-set.html

A SortedSet, that is, has the following operations in addition to Set operations:

- Range View. A view of a range of element within the SortedSet.
- Endpoints. The first or last element of the SortedSet.
- Compartor access. Returns the comparator, if any, of the SortedSet.

Range View Operations

Range-view operations write back to the backing sorted set and vice versa.

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/collections/interfaces/sorted-set.html

Sorted Sets have three range-view operations.

1. subset. This takes two end points, like subList (with lists). The endpoints must be objects that are comparable to elements in the sorted sort, not indices. Matching occurs that is case sensitive. A subset is half open, including the low endpoint but excluding the high endpoint. Matching occurs as if the supplied searching strings are looking for elements ordered in a dictionary.

```
private static SortedSet<String> mSortedSet = new TreeSet<String>(
        Arrays.asList("Xeno", "Marx", "Alphie", "Zara", "Marx", "Marx", "Hasselhoff",
                      "Beiber"));

static private void rangeViewsDemo() {
    System.out.println(mSortedSet);
    // Matching is case sensitive. Return half open interval.
    SortedSet<String> sortedSetRangeView = mSortedSet.subSet("B", "M");
    System.out.println(sortedSetRangeView);
}

// Output:
[Alphie, Beiber, Hasselhoff, Marx, Xeno, Zara]
[Beiber, Hasselhoff] // "Beiber" comes after "B", "Hasselhoff" comes before "M"

SortedSet<String> sortedSetRangeView = mSortedSet.subSet("B", "H");
// Returns
[Beiber] // "Beiber" comes after "B", "Hasselhoff" comes after "H" (and so it is excluded)
```

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/collections/interfaces/sorted-set.html
C:\Users\John\Documents\Sda\Code\Java\Examples\TutorialAtOracle\HelloWorld\src\helloworld\demo\collections\SortedSetDemo.java

To return a closed interval, and interval which includes both endpoints, request a subset from lowEndpoint to the successor to the highEndpoint. To specify the successor to the

highEndpoint apped "\0" (the null character) to the highEndpoint. But the highEndpiont must be a full match, not a partial match.

```
private static SortedSet<String> mSortedSet = new TreeSet<String>(
        Arrays.asList("Xeno", "Marx", "Alphie", "Zara", "Marx", "Marx", "Hasselhoff",
                      "Beiber"));

static private void subsetClosedIntervalDemo() {
    System.out.println(mSortedSet);
    // Matching is case sensitive.
    SortedSet<String> sortedSetRangeView = mSortedSet.subSet("B", "Marx\0");
    System.out.println(sortedSetRangeView);
}

// Output
[Alphie, Beiber, Hasselhoff, Marx, Xeno, Zara]
[Beiber, Hasselhoff, Marx]
```

For an open interval, which contains neither endpoint, specify the successor to the lowEndpoint and the highEndpoint. As above the lowEndpoint must be full match, not a partial match.

```
private static SortedSet<String> mSortedSet = new TreeSet<String>(
        Arrays.asList("Xeno", "Marx", "Alphie", "Zara", "Marx", "Marx", "Hasselhoff",
                      "Beiber"));

static private void subsetOpenIntervalDemo() {
    System.out.println(mSortedSet);
    // Matching is case sensitive.
    SortedSet<String> sortedSetRangeView = mSortedSet.subSet("Beiber\0", "M");
    System.out.println(sortedSetRangeView);
}

// Output
[Alphie, Beiber, Hasselhoff, Marx, Xeno, Zara]
[Hasselhoff]
```

2. headSet. Returns the initial portion of backing SortedSet up until, and excluding the object passed to headSet. The Null character "\0" can be used to include or exclude the object.
3. tailSet. Returns last portion of the backing SortedSet, including the object passed to the tailSet. The Null character "\0" can be used to include or exclude the object.

```
private static SortedSet<String> mSortedSet = new TreeSet<String>(
        Arrays.asList("Xeno", "Marx", "Alphie", "Zara", "Marx", "Marx", "Hasselhoff",
                      "Beiber"));

static private void tailAndHeadSetDemo() {
    System.out.println(mSortedSet);
    // Matching is case sensitive.
    SortedSet<String> sortedSetRangeView = mSortedSet.headSet("H");
    System.out.println(sortedSetRangeView);
}

// Output
[Alphie, Beiber, Hasselhoff, Marx, Xeno, Zara]
[Alphie, Beiber]

// Full word, exclude endpoint
SortedSet<String> sortedSetRangeView = mSortedSet.headSet("Hasselhoff");
// Returns
[Alphie, Beiber]

// Full word, include endpoint.
SortedSet<String> sortedSetRangeView = mSortedSet.headSet("Hasselhoff\0");
// Returns
```

```
[Alphie, Beiber, Hasselhoff]

// First Letter, includes all elements up until the non existant object "G".
SortedSet<String> sortedSetRangeView = mSortedSet.headSet("G");
// Returns
[Alphie, Beiber]

// First Letter, includes all elements up until the non existant object "G".
SortedSet<String> sortedSetRangeView = mSortedSet.headSet("H");
// Returns
[Alphie, Beiber]

// First Letter, include endpoint - doesn't include all.
SortedSet<String> sortedSetRangeView = mSortedSet.headSet("H\0");
// Returns
[Alphie, Beiber]
```

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/collections/interfaces/sorted-set.html
C:\Users\John\Documents\Sda\Code\Java\Examples\TutorialAtOracle\HelloWorld\src\helloworld\demo\collections\SortedSetDemo.java

Endpoint Operations

Retrieve the first and last element of a SortedSet with first() and last().

```
private static SortedSet<String> mSortedSet = new TreeSet<String>(
        Arrays.asList("Xeno", "Marx", "Alphie", "Zara", "Marx", "Marx", "Hasselhoff",
                      "Beiber"));

static private void endpointDemo() {
    System.out.println(mSortedSet);
    String endpoint = mSortedSet.last();
    System.out.println(endpoint);
}

// Output
[Alphie, Beiber, Hasselhoff, Marx, Xeno, Zara]
Zara
```

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/collections/interfaces/sorted-set.html
C:\Users\John\Documents\Sda\Code\Java\Examples\TutorialAtOracle\HelloWorld\src\helloworld\demo\collections\SortedSetDemo.java

You can use last() to iterate backwards. This is not recommended because it is an expensive operation.

```
static private void iterateBackwards() {
    System.out.println(mSortedSet);
    SortedSet<String> sortedSetRangeView = mSortedSet.headSet("Marx\0");
    // Iterate forward
    for (String s : sortedSetRangeView) {
        System.out.println(s);
    }
    System.out.println();
    // Iterate backwards
    Object object = sortedSetRangeView.last();
    for (int i = sortedSetRangeView.size(); i >= 1 ; i--) {
        System.out.println(object.toString());
        if (i > 1) {
            object = sortedSetRangeView.headSet(object.toString()).last();
        }
    }
}

// Output
[Alphie, Beiber, Hasselhoff, Marx, Xeno, Zara]
Alphie
Beiber
Hasselhoff
Marx
```

```
Marx
Hasselhoff
Beiber
Alphie
```

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/collections/interfaces/sorted-set.html
C:\Users\John\Documents\Sda\Code\Java\Examples\TutorialAtOracle\HelloWorld\src\helloworld\demo\collections\SortedSetDemo.java

Comparator Accessor

The SortedSet interface contains an accessor method called comparator that returns the Comparator used to sort the set, or null if the set is sorted according to the natural ordering of its elements. This method is provided so that sorted sets can be copied into new sorted sets with the same ordering.

See "SortedSet conversion constructor" in Intro

SortedMap Interface

A SortedMap is a Map with elements in ascending order. The order will be the key's natural ordering or according to a Comparator (supplied at the time of the SortedMap creation).

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/collections/interfaces/sorted-map.html

The SortedMap Interface:

```
public interface SortedMap<K, V> extends Map<K, V>{
    Comparator<? super K> comparator();
    SortedMap<K, V> subMap(K fromKey, K toKey);
    SortedMap<K, V> headMap(K toKey);
    SortedMap<K, V> tailMap(K fromKey);
    K firstKey();
    K lastKey();
}
```

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/collections/interfaces/sorted-map.html

... That is, the SortedMap provides additional operations, analogous to the operations SortedSet provides above a Set:

- Range view — performs arbitrary range operations on the sorted map
- Endpoints — returns the first or the last key in the sorted map
- Comparator access — returns the Comparator, if any, used to sort the map

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/collections/interfaces/sorted-map.html

Because this interface is a precise Map analog of SortedSet, all the idioms and code examples in The SortedSet Interface section apply to SortedMap with only trivial modifications.

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/collections/interfaces/sorted-map.html

Implementations

Summary

"Implementations" implement an interface. You can't use an interface without specifying an implementation.

```
// The Map interface is implemented here with a HashMap
// (the common implementation for a Map)
Map<String, Integer> map = new HashMap<String, Integer>();
```

In general you only have to think about the implementation when you create the collection (as above). Choice of implementation is mainly an issue of performance (although sometimes you choose an implementation for behavioral reason such as ordering). Given that the implementation is only specified at creation, the programmer is free to change the implementation if different (performance or behavioral) goals emerge.

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/collections/implementations/index.html

There are the following kinds of implementation:

- **General purpose**. For typical uses.
- **Commonly used**. For the most typical use.
- **Special purpose**. For uncommon behaviour (e.g. insertion order rather than natural order, restricting use, etc) or performance characteristics.
- **Concurrent**. Supports concurrency, typically at the expense of single threaded performance. Found in java.util.concurrent.
- **Wrapper**. Used in combination with other implementations (often general purpose) for added or restricted functionality.
- **Convenience**. Mini-implementations serving has handy alternatives to general purpose or special purpose implementations.
- **Abstract**. Facilitates custom collection implementations. Rarely needed.

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/collections/implementations/index.html

General-Purpose and commonly used implementations. Commonly used implementations marked with "*". In addition to the implementations listed below Queue has a PriorityQueue implementation which orders elements according to their value.

Table 1 General-Purpose and commonly used implementations

Interfaces	Hash table Implementations	Resizable array Implementations	Tree Implementations	Linked list Implementations	Hash table + Linked list Implementations
Set	HashSet * (No order)		TreeSet (natural order)		LinkedHashSet (Insertion order)
SortedSet			TreeSet*		
List		ArrayList *		LinkedList	
Queue				LinkedList (FIFO)	
Deque		ArrayDeque		LinkedList	

Map	**HashMap** *				LinkedHashMap
SortedMap			**TreeMap***		

(Oracle, 2017) *http://docs.oracle.com/javase/tutorial/collections/implementations/index.html*

Collection Implementations

Collection and Implementation	**Peformance**	**Ordering**	**Other**
Set			
HashSet	Best performing	None	
TreeSet	Substantially slower than HashSet	On values.	
LinkedHashSet	Slightly worse than HashSet	Insertion order.	
EnumSet			Provides for range operations on Enums and Bit flag operation replacment
CopyOnWriteArraySet			Well suited to maintaining event-handler lists that must prevent duplicates.
Map			
HashMap	Best performing	None.	
TreeMap		Natural ordering.	
LinkedHashMap	Near HashMap performance	Insertion Order.	Merely looking up the value associated with a key brings that key to the end of the map.
EnumMap			A map implementation for use with enums
WeakHashMap			
IdentityHashMap			
List			
ArrayList	Best performing (constant-time)		
LinkedList	Preforms well if frequently: * Add elements to beginning; or * Delete elements from interior.		ArrayList usually faster.
CopyOnWriterArrayList			
Queue			
PriorityQueue			
Deque			

LinkedList			
ArrayDeque			
SortedSet			
TreeSet	Substantially slower than HashSet	On values.	Only implemenetation for Sorted Set
Map			
HashMap			(Analogous to HashSet)
TreeMap			(Analogous to TreeSet)
LinkedHashMap			(Analogous to LinkedHashSet)

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/collections/interfaces/map.html
http://docs.oracle.com/javase/tutorial/collections/interfaces/set.html

All of these generally purpose implementations have the following features:

- Permit null elements, keys, and values.
- Are not synchronized (they are not thread-safe).
- Serializable.
- Support a public clone method.
- Fast-fail iterators. That is, when an illegal concurrent modification during iteration occurs the iteration will fail cleanly and quickly.

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/collections/implementations/index.html

If you want a concurrent implementation either:

- Use a synchronization wrapper implementations, which can make any collection synchronized; or
- Use implementations from java.util.concurrent (BlockingQueue or ConcurrentMap) which provide "higher [?] concurrency than mere [?] synchronized implementations".

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/collections/implementations/index.html

Set Implementations

General

HashSet Notes. Iteration cost is linear but the sum of the number of entries and the number of empty slots (as defined by capacity). The default capacity is 16. As a general rule choose a capacity twice the size you expect the set to grow to.

```
Set<String> s = new HashSet<String>(64);
```

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/collections/implementations/set.html

EnumSet Range

The special purpose set implementation, EnumSet, allows you to iterate over a range of elements within a set.

```
private enum Day {
    SUNDAY,
    MONDAY,
    TUESDAY,
    WEDNESDAY,
    THURSDAY,
    FRIDAY,
    SATURDAY
}

static private void EnumSetImplementation() {
    for (Day d : Day.values()){
        System.out.println(d);
    }

    System.out.println();
    EnumSet<Day> Weekdays = EnumSet.range(Day.MONDAY, Day.FRIDAY);
    for (Day d: Weekdays) {
        System.out.println(d);
    }
}

// Output
SUNDAY
MONDAY
TUESDAY
WEDNESDAY
THURSDAY
FRIDAY
SATURDAY

MONDAY
TUESDAY
WEDNESDAY
THURSDAY
FRIDAY
```

(Oracle, 2017) *http://docs.oracle.com/javase/tutorial/collections/implementations/set.html*

EnumSet Bit flag Replacment

EnumSets afford a typesafe replacement for bit flag operations.

```
// helloworld/Style.java
public enum Style {
    BOLD,
    ITALIC,
    UNDERLINE,
    STRIKETHROUGH
}

// helloworld/Word.java
public class Word {
    public String content = "";
    public EnumSet<Style> styleEnumSet = null;

    public Word(String content, EnumSet<Style> styleApplied) {
        this.content = content;
        this.styleEnumSet = styleApplied;

    }

    public String toString() {
        for (Style style : styleEnumSet) {
            switch (style) {
                case BOLD:
                    content = "<strong>" + content + "</strong>";
```

```
                    break;
                case ITALIC:
                    content = "<em>" + content + "</em>";
                    break;
                case STRIKETHROUGH:
                    content = "<del>" + content + "</del>";
                    break;
                default:
                    try {
                        throw new Exception("Unexpected case in switch");
                    } catch (Exception e) {
                        e.printStackTrace();
                    }
            }
        }

        return content;
    }
}

// helloworld/demo/collections/ImplementationsDemo.java
static private void EnumSetAsBitFlags() {
    EnumSet<Style> myStyle = EnumSet.of(Style.BOLD, Style.ITALIC);
    Word myHeading = new Word("Cool", myStyle);
    System.out.println(myHeading);
}

// Output
<em><strong>Cool</strong></em>
```

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/collections/implementations/set.html
C:\Users\John\Documents\Sda\Code\Java\Examples\TutorialAtOracle\HelloWorld\src\helloworld\demo\collections\ImplementationsDemo.java

CopyOnWriteArraySet

See ...

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/collections/implementations/set.html

Map Implementations

Limit Entries

To limit the entries in the map override removeEldestEntry() of a LinkedHashMap.

```
static private Map<String, Integer> getFrequencyTableMap() {
    String[] keys = {
            "Xeno", "Marx", "Alphie", "Zara", "Marx", "Marx", "Hasselhoff", "Beiber"
    };

    Map<String, Integer> map = new LinkedHashMap<String, Integer>() {
        private static final int MAX_ENTRIES = 5;

        protected boolean removeEldestEntry(Map.Entry eldest) {
            return size() > MAX_ENTRIES;
        }
    };

    Integer value = null;
    for (String key : keys) {
        // Check if key already in map
        Integer priorFrequency = map.get(key);
        Integer frequency = (priorFrequency == null) ? 1 : priorFrequency + 1;
        map.put(key, frequency);
    }

    return map;
```

```
}

static private void mapImplementation() {
    Map<String, Integer> map = getFrequencyTableMap();
    System.out.println(map);
    map.put("Fonzi", 12);
    System.out.println(map);
}

// Output
{Marx=3, Alphie=1, Zara=1, Hasselhoff=1, Beiber=1}
{Alphie=1, Zara=1, Hasselhoff=1, Beiber=1, Fonzi=12}
```

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/collections/implementations/map.html
C:\Users\John\Documents\Sda\Code\Java\Examples\TutorialAtOracle\HelloWorld\src\helloworld\demo\collections\MapDemo.javas

EnumMap

EnumMap is for use with enum keys. If you want to map an enum to a value you should always use an EnumMap in preference to an array.

```
private enum Folk {
    XENO,
    MARX,
    APLPHE,
    ZARA,
    HASSELHOFF,
    BEIBER
}

static private void enumMapDemo() {
    Map<Folk, Integer> coolnessMap = new EnumMap<Folk, Integer>(Folk.class);

    coolnessMap.put(Folk.XENO, 12);
    coolnessMap.put(Folk.MARX, 3);
    coolnessMap.put(Folk.APLPHE, 2);
    coolnessMap.put(Folk.ZARA, 7);
    coolnessMap.put(Folk.HASSELHOFF, 2);
    coolnessMap.put(Folk.BEIBER, 24);

    for (Map.Entry<Folk, Integer> e : coolnessMap.entrySet()) {
        System.out.println(e.getKey() + ": " + e.getValue());
    }
    System.out.println();
    System.out.println(coolnessMap);
}

// Output
XENO: 12
MARX: 3
APLPHE: 2
ZARA: 7
HASSELHOFF: 2
BEIBER: 24

{XENO=12, MARX=3, APLPHE=2, ZARA=7, HASSELHOFF=2, BEIBER=24}
```

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/collections/implementations/map.html
C:\Users\John\Documents\Sda\Code\Java\Examples\TutorialAtOracle\HelloWorld\src\helloworld\demo\collections\MapDemo.javas

Other Special Purpose Map Implementations

WeakHashMap stores a weak reference to keys. When the key is no longer referenced outside of the WeakHashMap the key-value pair is garbage collected. Good for registery-like structures.

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/collections/implementations/map.html

IdentityHashMap. Good for:

- Serialization;
- Deep copying;
- Thwarting spoof attacks (as it never invokes the equals method on its keys).

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/collections/implementations/map.html

Queue Implementations

The PriorityQueue implementation (of a Queue) provide and iterator but this iterator is not guaranteed to traverse the elements in order. Instead consider Arrays.sort(pq.toArray()).

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/collections/implementations/map.html

Deque Implementations

ArrayDeque:

- Resizable;
- Doesn't allow nulls;
- More efficient for add and remove operations at both ends.

LinkedList:

- More flexible (than ArrayDeque);
- Implements list operations;
- Allows nulls;
- Less efficient for add and remove operations at both ends.
- Good for removing the current element during iteration, but not ideal for iteration.

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/collections/implementations/deque.html

Wrapper Implementations

There are three wrapper implementations for collections:

- Synchronization wrappers;
- Unmodiffiable wrappers; and
- Checked Interface wrappers.

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/collections/implementations/wrapper.html

Synchronization wrappers

The syncrhonization wrappers add automatic synchronization (thread-safety) to a collection.

```
public static <T> Collection<T> synchronizedCollection(Collection<T> c);
public static <T> Set<T> synchronizedSet(Set<T> s);
public static <T> List<T> synchronizedList(List<T> list);
public static <K,V> Map<K,V> synchronizedMap(Map<K,V> m);
public static <T> SortedSet<T> synchronizedSortedSet(SortedSet<T> s);
public static <K,V> SortedMap<K,V> synchronizedSortedMap(SortedMap<K,V> m);
```

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/collections/implementations/wrapper.html

To guarantee serial access, all access to the backing collection must be accomplished through the returned collection. To do this do not keep a reference to the backing collection.

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/collections/implementations/wrapper.html

Create a synchronized collection like this ...

```
List<Type> list = Collections.synchronizedList(new ArrayList<Type>());
```

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/collections/implementations/wrapper.html

Iterate over a wrapper-synchronized collection.

```
Collection<Type> c = Collections.synchronizedCollection(myCollection);
synchronized(c) {
    for (Type e : c)
        foo(e);
}
```

If an explicit iterator is used, the iterator method must be called from within the synchronized block.

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/collections/implementations/wrapper.html

When Iterating over a synchronized map ensure that you synchronize in the map itself rather than one of its collection views.

```
Map<KeyType, ValType> m = Collections.synchronizedMap(new HashMap<KeyType, ValType>());
    ...
Set<KeyType> s = m.keySet();
    ...
// Synchronizing on m, not s!
synchronized(m) {
    while (KeyType k : s)
        foo(k);
}
```

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/collections/implementations/wrapper.html

Unmodifiable Wrappers

Unmodifiable wrappers make collections read only.

```
public static <T> Collection<T> unmodifiableCollection(Collection<? extends T> c);
public static <T> Set<T> unmodifiableSet(Set<? extends T> s);
public static <T> List<T> unmodifiableList(List<? extends T> list);
public static <K,V> Map<K, V> unmodifiableMap(Map<? extends K, ? extends V> m);
public static <T> SortedSet<T> unmodifiableSortedSet(SortedSet<? extends T> s);
public static <K,V> SortedMap<K, V> unmodifiableSortedMap(SortedMap<K, ? extends V> m);
```

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/collections/implementations/wrapper.html

Checked Interface Wrappers

When working with a generic collection checked interface wrappers return a dynamically type-safe view of the specified collection. A ClassCastExceptio is thrown if client code tries to add and element of the wrong type.

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/collections/implementations/wrapper.html

Convenience Implementations

See Array Translations.

nCopies for Lists

Using the immutable multiple copy list ncopies() for creating lists.

```
static public void creation() {
    List<String> folk = new ArrayList<String>(Collections.nCopies(5, (String)null));
    System.out.println(folk);

    folk.add("Paul");
    System.out.println(folk);
    folk.add(0, "John");
    folk.add(2, "Ringo");
    System.out.println(folk);
}

// Output
[null, null, null, null, null]
[null, null, null, null, null, Paul]
[John, null, Ringo, null, null, null, null, Paul]
```

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/collections/implementations/convenience.html
C:\Users\John\Documents\Sda\Code\Java\Examples\TutorialAtOracle\HelloWorld\src\helloworld\demo\collections\ListDemo.java

You can also append a number of copies to the end of a list.

```
lovablePets.addAll(Collections.nCopies(69, "fruit bat"));
```

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/collections/implementations/convenience.html

Use the indexed version of .addAll() to insert a number of copies into the middle of a list.

Singleton Set

A singleton set returns a set with one element.

Useful for removing all elements from a collection.

```
c.revomveAll(Collections.singleton(e));
```

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/collections/implementations/convenience.html

Useful for removing all elements in a map with a particular value.

```
jobMap.values().removeAll(Collections.singleton(LAWYER));
```

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/collections/implementations/convenience.html

Empty Collections

You can use methods in Collections to return an empty collection for a set, list, and map: emptySet(), emptyList(), and emptyMap(). Useful for input to methods that take a collection of values, and you don't want to provide any values.

```
tourist.declarePurchases(Collections.emptySet());
```

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/collections/implementations/convenience.html

Algorithms

Intro

"Algorithms", in the collections context, are methods that perform operation on a Collection. They are polymorphic in that they operate regardless of the implementation. All of these methods come from the Collections class. Most of these methods operate on List instances but some operate on arbitrary Collection instances.

(Oracle, 2017) https://docs.oracle.com/javase/tutorial/collections/algorithms/index.html

Ordering / sorting

How to sort a collection

Generally the collection that you sort is a List collection (?).

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/collections/interfaces/order.html

A collection can be sorted in three ways.

1. By using a collection implementation that preserves sorting.

```
static private void frequencyTable(){
    ...
    // HashMap is most efficient.
//  Map<String, Integer> map = new HashMap<String, Integer>();

    // A tree map preserves key order
    Map<String, Integer> map = new TreeMap<String, Integer>();    ...
}

// Output
6 distinct words:
{Alphie=1, Beiber=1, Hasselhoff=1, Marx=3, Xeno=1, Zara=1}
```

C:\Users\John\Documents\Sda\Code\Java\Examples\TutorialAtOracle\HelloWorld\src\helloworld\demo\collections\MapDemo.java

2. By using a collection interface specifically designed for sorting.

```
// A tree map preserves key order
SortedMap<String, Integer> map = new TreeMap<String, Integer>();
```

C:\Users\John\Documents\Sda\Code\Java\Examples\TutorialAtOracle\HelloWorld\src\helloworld\demo\collections\MapDemo.java

3. By using Collections.sort() on a List.

```
private static List<String> mList = Arrays.asList("Xeno",
                                                   "Marx",
                                                   "Alphie",
                                                   "Zara",
                                                   "Marx",
                                                   "Marx",
                                                   "Hasselhoff",
                                                   "Beiber");

static public void start() {
    System.out.println(mList);
    Collections.sort(mList);
    System.out.println(mList);
}

// Output
[Xeno, Marx, Alphie, Zara, Marx, Marx, Hasselhoff, Beiber]
[Alphie, Beiber, Hasselhoff, Marx, Marx, Marx, Xeno, Zara]
```

The sorting algorithm used is an optimized merge sort. This is fast and stable. Stable in the sense that it doesn't reorder equal elements (which may be important in your application. E.g. An email inbox).

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/collections/interfaces/order.html
C:\Users\John\Documents\Sda\Code\Java\Examples\TutorialAtOracle\HelloWorld\src\helloworld\demo\collections\OrderingDemo.java

The comparable interface

When a collection, like a list, is sortable it is because the type of element the collection stores implements a comparable interface. A comparable specifies the "natural ordering" of the type. For example a String type implements a comparable that specfies an alphabetic ordering as "natural". So ording a List<String> will result an an alphabetically arranged list.

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/collections/interfaces/order.html

The natural ordering of common Java types:

Class	**Natural Ordering**
Byte	Signed numerical
Character	Unsigned numerical
Long	Signed numerical
Integer	Signed numerical
Short	Signed numerical
Double	Signed numerical
Float	Signed numerical
BigInteger	Signed numerical
BigDecimal	Signed numerical
Boolean	Boolean.FALSE < Boolean.TRUE
File	System-dependent lexicographic on path name

String	Lexicographic
Date	Chronological
CollationKey	Locale-specific lexicographic

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/collections/interfaces/order.html

Custom comparables

Generally you can rely on the inbuilt comparables of the common types. That is, you will rarely need to write your own, custom, comparable. But if you do then this section is where the action is at.

To write your own comparable:

Implement the comparable interface.

```
public interface Comparable<T> {
    public int compareTo(T o);
}
```

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/collections/interfaces/order.html

Implement the compareTo() method. Return the following results:

Receiving object compared to specified object	**Return int**
Less than	Negative integer
Equals	0
Greate than	Postive integer.

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/collections/interfaces/order.html

Custom comparable example.

```
public class Name implements Comparable<Name> {
    private final String firstName, lastName;

...

    public int compareTo(Name n) {
        int lastCmp = lastName.compareTo(n.lastName);
        return (lastCmp != 0 ? lastCmp : firstName.compareTo(n.firstName));
    }
}

// Usage
public class NameSort {
    public static void main(String[] args) {
        Name nameArray[] = {
            new Name("John", "Smith"),
            new Name("Karl", "Ng"),
            new Name("Jeff", "Smith"),
            new Name("Tom", "Rich")
        };

        List<Name> names = Arrays.asList(nameArray);
        Collections.sort(names);
        System.out.println(names);
    }
}
```

```
// Output
[Karl Ng, Tom Rich, Jeff Smith, John Smith]
```

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/collections/interfaces/order.html

There are four restrictions on the behaviour of the compareTo() method.

See (Oracle, 2017) http://docs.oracle.com/javase/7/docs/api/java/lang/Comparable.html

Comparators

There might be times when you want to specify an ordering from outside the class you wish to order. This could occur if you want to either: override the ordering specified within a class (because the class implements a comparable); or to specify an ordering for a class that has no ordering (because the class doesn't implement a comparable) and you don't want to define this ordering internal to the class. To effect this external specification of ordering you can implement a comparator interface.

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/collections/interfaces/order.html

The comparator interface.

```
public interface Comparator<T> {
    int compare(T o1, T o2);
}
```

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/collections/interfaces/order.html

Implement a comparator interface (and the compare method), example.

```
// Existing class
public class Employee implements Comparable<Employee> {
    public Name name()     { ... }
    public int number()    { ... }
    public Date hireDate() { ... }
       ...
}

public class EmpSort {
    // Implement the Comparator Interface and the compare method
    static final Comparator<Employee> SENIORITY_ORDER =
                                        new Comparator<Employee>() {
            public int compare(Employee e1, Employee e2) {
                return e2.hireDate().compareTo(e1.hireDate());

               // Don't do this!!
               // return -r1.hireDate().compareTo(r2.hireDate());
            }
    };

    // Fetch a Collection of Employees (e.g. from a database)
    static final Collection<Employee> employees = ... ;

    public static void main(String[] args) {
        // Convert collection to a List.
        List<Employee> e = new ArrayList<Employee>(employees);

        // Perform sort, passing the comparator as an argument.
        // The sort therefore uses the comparator, not the comparable (implemented in the
        // Employee class).
        Collections.sort(e, SENIORITY_ORDER);
        System.out.println(e);
    }
}
```

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/collections/interfaces/order.html

Always design your comparator so that it returns the same result as equals. To do this always return the same ordering, for the same elements.

The Comparator in the preceding program works fine for sorting a List, but it does have one deficiency: It cannot be used to order a sorted collection, such as TreeSet, because it generates an ordering **that is not compatible with equals**. This means that this Comparator equates objects that the equals method does not. In particular, any two employees who were hired on the same date will compare as equal. When you're sorting a List, this doesn't matter; but when you're using the Comparator to order a sorted collection, it's fatal. If you use this Comparator to insert multiple employees hired on the same date into a TreeSet, only the first one will be added to the set; the second will be seen as a duplicate element and will be ignored.

To fix this problem, simply tweak the Comparator so that it produces an ordering that is compatible with equals. In other words, tweak it so that the only elements seen as equal when using compare are those that are also seen as equal when compared using equals. The way to do this is to perform **a two-part comparison** (as for Name), where the first part is the one we're interested in — in this case, the hire date — and the second part is an attribute **that uniquely identifies the object**. Here the employee number is the obvious attribute. This is the Comparator that results.

```
static final Comparator<Employee> SENIORITY_ORDER =
                                        new Comparator<Employee>() {
    public int compare(Employee e1, Employee e2) {
        int dateCmp = e2.hireDate().compareTo(e1.hireDate());
        if (dateCmp != 0)
            return dateCmp;

        return (e1.number() < e2.number() ? -1 :
                (e1.number() == e2.number() ? 0 : 1));
    }
};
```

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/collections/interfaces/order.html

Shuffling

The Shuffle algorithm (Collections.shuffle()) randomizes element order. Uses a fair source of randomness. You can optionally suppy a source of randomness, a Random object.

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/collections/interfaces/order.html
(Oracle, 2017) https://docs.oracle.com/javase/tutorial/collections/interfaces/list.html#shuffle

Routine Data Manipulation

Five other algorithms in Collections include:

- reverse - reverses the order of elements of a List.
- fill - overwrites every element in a List with a specified value.
- copy - take a source and destination List. Copies elements of source into destination, overwriting destination contents. The destination List must be at least as long as the source. If the destination list is longer then the remaining destination elements are unaffected.
- swap - swaps List elements as the specified positions.
- addAll - adds the supplied elements to a Collection. You can add the elements individually or as an array.

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/collections/interfaces/order.html

Searching

binarySearch() operates on a List that has been sorted ascending, either by natural ordering or with a comparator. The return value will be:

- If found: index; or
- If not found: (-(insertion point) - 1), where insertion point is the point at which the value would be inserted into the List (or index of the first element greater than list.Size() if all the elements are less than the search element).

(Oracle, 2017) https://docs.oracle.com/javase/tutorial/collections/algorithms/index.html

Composition

frequency() - counts the number of times an element occurs in a collection.

```
static private void frequencyDemo() {
    List<String> list = new ArrayList<String>();
    Collections.addAll(list, "Xeno", "Marx", "Alphie", "Zara", "Marx", "Marx",
                       "Hasselhoff", "Beiber");
    String element = "Marx";
    int frequency = Collections.frequency(list, element);

    System.out.println(list);
    System.out.format("Frequency of \"%s\" is %d", element, frequency);
}

// Output
[Xeno, Marx, Alphie, Zara, Marx, Marx, Hasselhoff, Beiber]
Frequency of "Marx" is 3
```

(Oracle, 2017) https://docs.oracle.com/javase/tutorial/collections/algorithms/index.html
C:\Users\John\Documents\Sda\Code\Java\Examples\TutorialAtOracle\HelloWorld\src\helloworld\demo\collections\AlgorithmsDemo.java

disjoint() - a boolean that returns true if two collections contain no common element.

```
static private void disjointDemo() {
    List<String> alphaList = new ArrayList<String>();
    Collections.addAll(alphaList, "Xeno", "Marx", "Alphie", "Zara", "Marx", "Marx",
                       "Hasselhoff", "Beiber");

    List<String> betaList = new ArrayList<>();
    Collections.addAll(betaList, "Lisa", "Sarah", "Fonz");

    Boolean isDisjoint = Collections.disjoint(alphaList, betaList);
    System.out.println(alphaList);
    System.out.println(betaList);

    System.out.format("Lists are disjoint: %b", isDisjoint);
}

// Output
[Xeno, Marx, Alphie, Zara, Marx, Marx, Hasselhoff, Beiber]
[Lisa, Sarah, Fonz]
Lists are disjoint: true
```

(Oracle, 2017) https://docs.oracle.com/javase/tutorial/collections/algorithms/index.html
C:\Users\John\Documents\Sda\Code\Java\Examples\TutorialAtOracle\HelloWorld\src\helloworld\demo\collections\AlgorithmsDemo.java

Min and Max

min() and max() return the first and last element in a collection, if the collection was ordered. Optionally takes a comparator.

```
static private void extremeDemo() {
    List<String> list = new ArrayList<String>();
    Collections.addAll(list, "Xeno", "Marx", "Alphie", "Zara", "Marx", "Marx",
                       "Hasselhoff", "Beiber");
    System.out.println(list);
    System.out.println("max: " + Collections.max(list));
    System.out.println("min: " + Collections.min(list));
}

// Output
```

(Oracle, 2017) https://docs.oracle.com/javase/tutorial/collections/algorithms/index.html
C:\Users\John\Documents\Sda\Code\Java\Examples\TutorialAtOracle\HelloWorld\src\helloworld\demo\collections\AlgorithmsDemo.java

XML Processing (JAXP)

Overview

Java has a built-in "Java API for XML processing": JAXP. This API provides XML processing and parsing through:

- SAX: "Simple API for XML Parsing";
- DOM: The "Document Object Model";
- XSL: XSLT "Extensible Stylesheet Transformations" and XPath (the querying language for selecting nodes in an XML document); and
- StAX: "Streaming API for XML".

(Oracle, 2017) *http://docs.oracle.com/javase/tutorial/jaxp/intro/index.html*

JAXP allows you to use whatever XML parser, whatever implementation of SAX, DOM, or XSL you like, through a "pluggability layer".

(Oracle, 2017) *http://docs.oracle.com/javase/tutorial/jaxp/intro/index.html*

The componenets of JAXP:

	SAX	**DOM**	**StAX**	**XSL (XSLT and XPath)**
Java Packages	javax.xml.parsers org.xml.sax	javax.xml.parsers	javax.xml.stream	javax.xml.transform
Ease of Use	Medium	High	High	
CPU and Memory Efficientcy	Good	Varies	Good	
XPath Capability	No	Yes	No	
Read XML	Yes	Yes	Yes	Yes
Write XML	No	Yes	Yes	Yes
Create, Read, Update, Delete	Yes	Yes	No	
Forward Only	Yes	No	Yes	
Supports XML Schema	Yes	Yes	Yes	

(Oracle, 2017) *http://docs.oracle.com/javase/tutorial/jaxp/intro/package.html*
http://docs.oracle.com/javase/tutorial/jaxp/stax/why.html

Other APIs in the wild: JDOM (http://www.jdom.org) and DOM4J (http://dom4j.sourceforge.net/). These are object-oriented.

Differences between DOM and the third party API JDOM and DOM4J:

- DOM is primarly for documents (often containing mixed content). JDOM and DOM4J are primarly for non document structured data (where an element contains either text or child elements, but not both).
- JDOM and DOM4J are object oriented and are designed from the ground up to work with Java.
- DOM (and SAX) support XML Schmea, JDOM and DOM4J do not (confirm?).
- DOM, although complex, will "handle virtually anything you through at it".

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/jaxp/dom/when.html

JAXP versions are identified at https://jaxp.java.net/. At the moment these entail:

JAXP Version	Java SE (Oracle)	Open JDK
1.4	Java SE 6.0	OpenJDK7
1.5	Java SE 8.0	OpenJDK7 update 40
1.6	Java SE 8.0	OpenJDK 8

https://jaxp.java.net/

JAXP official documentation: ...

(Oracle, 2017) "Java API for XML Processing (JAXP)" http://docs.oracle.com/javase/tutorial/jaxp/TOC.html

JAXP Samples that accompany the documentation.

"JAXP-Sources. Samples" https://java.net/projects/jaxp-sources/sources/svn/show/trunk/samples

Simple API for XML (SAX)

SAX Overview

SAX basically works by your implementation of document events (arising from a "ContentHandler") which the Sax Paser/XML reader raises. Those document events include startDocument, endDocument, startElement (which includes attributes), endElement.

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/jaxp/intro/simple.html

SAX Coding procedure

Sax Read XML

Setup and call Sax Parser and XML Reader of the Sax Parser. We feed a file and a custom class that subclasses org.xml.sax.helpers.DefaultHandler.

```
import org.xml.sax.SAXException;
import org.xml.sax.XMLReader;

import javax.xml.parsers.ParserConfigurationException;
import javax.xml.parsers.SAXParser;
import javax.xml.parsers.SAXParserFactory;
import java.io.File;
import java.io.IOException;

/**
 * Call this from IntelliJ Run Configuration with a working directory set to
```

```
 * C:\Users\John\Documents\Sda\Code\BjaxXsl\Examples\XmlAviationWeather\XmlSource
 */
public class JaxpDemo {
    static public void main(String[] args) throws Exception {
        String argument = null;
        String fileName = null;

        // Display usage if there is more than one arg
        if (args.length != 1) {
            usage();
        } else {

            argument = args[0];

            // Display usage for any argument intended as a switch '-argument',
            if ((argument == null) || (argument.matches("-.*"))) {
                usage();

                // Else regard the argument as a fileName.
            } else {
                fileName = argument;
                setupAndCallSaxXmlReader(fileName);
            }
        }

    }

      static private String convertToPathSlashFileName(String fileName) {
        String pathSlashFileName = new File(fileName).getAbsolutePath();

        if (File.separatorChar != '/') {
            pathSlashFileName = pathSlashFileName.replace(File.separatorChar, '/');
        }

        if (!pathSlashFileName.startsWith("/")) {
            pathSlashFileName = "/" + pathSlashFileName;
        }

        return "file:" + pathSlashFileName;
    }

    static private void usage() {
        System.out.println("Usage: JaxpDemo <file.xml>");
        System.out.println("       -usage or -help: this message");
        System.out.println("       [But run from IntelliJ]");
        System.exit(1);
    }

    static private void setupAndCallSaxXmlReader(
            String fileName) throws ParserConfigurationException, SAXException,
            IOException {

        SAXParserFactory spf = SAXParserFactory.newInstance();
        spf.setNamespaceAware(true);
        SAXParser saxParser = spf.newSAXParser();

        XMLReader xmlReader = saxParser.getXMLReader();
        xmlReader.setContentHandler(new SaxLocalNameCount());
        xmlReader.parse(convertToPathSlashFileName(fileName));
    }
}
```

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/jaxp/sax/parsing.html
C:\Users\John\Documents\Sda\Code\BjaxXsl\Examples\XmlAviationWeather\JaxpDemo\src\JaxpDemo.java

In the SAX subclass of org.xml.sax.helpers.DefaultHandler implement various ContentHandler Document event callbacks.

```
import org.xml.sax.Attributes;
import org.xml.sax.SAXException;
import java.util.LinkedHashMap;
import java.util.Map;

public class SaxLocalNameCount extends org.xml.sax.helpers.DefaultHandler {
```

```
    private Map<String, Integer> mTags;

    @Override
    public void startDocument() throws SAXException {
        super.startDocument();

        mTags = new LinkedHashMap<String, Integer>();
    }

    @Override
    public void startElement(String uri, String localName, String qName,
                             Attributes attributes) throws SAXException {
        super.startElement(uri, localName, qName, attributes);

        String key = localName;
        Object value = mTags.get(key);

        if (value == null) {
            mTags.put(key, new Integer(1));
        } else {
            int count = ((Integer) value).intValue();
            count++;
            mTags.put(key, new Integer(count));
        }
    }


    @Override
    public void endDocument() throws SAXException {
        super.endDocument();

        for (Map.Entry<String, Integer> e : mTags.entrySet()) {
            System.out.format("Local Name '%s' occurs %d times.%n", e.getKey(),
                              e.getValue());
        }
    }
}
```

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/jaxp/intro/simple.html
(Oracle, 2017) http://docs.oracle.com/javase/tutorial/jaxp/sax/parsing.html
C:\Users\John\Documents\Sda\Code\BjaxXsl\Examples\XmlAviationWeather\JaxpDemo\src\SaxLocalNameCount.java

Results.

```
// Example input
<?xml version="1.0" ?>
<!--<?xml-stylesheet type="text/xsl" href="NameValuePairs.xslt" ?>-->
<takeOffAndLandingWeatherReports
xmlns="http://tempuri.org/TakeOffAndLandingWeatherReports.xsd"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://tempuri.org/TakeOffAndLandingWeatherReports.xsd
TakeOffAndLandingWeatherReports.xsd">

  <!-- A series of weather reports for various locations and date/times -->
  <report>
    <location>YSSY</location>
    <dateTimeUtc>2003-10-23T15:30:00Z</dateTimeUtc>

    <wind>
      <direction reference="magnetic">060</direction>
      <speed unit="knots">23</speed>
    </wind>

    <qnh unit="hectoPascals">1013.12</qnh>
    <temperature unit="celsius">25</temperature>
    <dewPoint unit="celsius">-02</dewPoint>

    <sky>
      <cloud cover="few (0-2 eights)">
        <height unit="feet" reference="agl">4000</height>
      </cloud>
```

```
      <cloud cover="broken (5-7 eights)">
        <height unit="feet" reference="agl">11000</height>
      </cloud>
      <ceiling cover="broken (5-7 eights)">
        <height unit="feet" reference="agl">4000</height>
      </ceiling>

      <visibility>Greater than 10 kilometres.</visibility>

      <phenomenon intensity="light" action="showers">rain</phenomenon>
      <phenomenon intensity="medium" action="thunderstorm">hail</phenomenon>
    </sky>
  </report>

  <report>
    <location>YMML</location>
...

// Example output
Local Name 'takeOffAndLandingWeatherReports' occurs 1 times.
Local Name 'report' occurs 3 times.
Local Name 'location' occurs 3 times.
Local Name 'dateTimeUtc' occurs 3 times.
Local Name 'wind' occurs 3 times.
Local Name 'direction' occurs 3 times
....
```

Add XML Error checking. You can set the error checking at two levels:

- Well-formed: there are no syntax errors in the document.
- Valid: The document is well formed and complies with the grammar defined in an XML Schema document (DTD validation can also be performed).

```
// CentralSaxErrorHandler.java
public class CentralSaxErrorHandler implements org.xml.sax.ErrorHandler {

    private PrintStream out;

    CentralSaxErrorHandler(PrintStream out) {
        this.out = out;
    }

    private String getParseExceptionInfo(SAXParseException spe) {
        String systemId = spe.getSystemId();

        if (systemId == null) {
            systemId = "null";
        }

        String info = String.format("URI=%s Line=%d : %s", systemId, spe.getLineNumber(),
                                    spe.getMessage());

        return info;
    }

    @Override
    public void warning(SAXParseException spe) throws SAXException {
        out.println("Warning: " + getParseExceptionInfo(spe));
    }

    @Override
    public void error(SAXParseException spe) throws SAXException {
        String message = "Error: " + getParseExceptionInfo(spe);
        throw new SAXException(message);
    }

    @Override
    public void fatalError(SAXParseException spe) throws SAXException {
        String message = "Fatal Error: " + getParseExceptionInfo(spe);
        throw new SAXException(message);
    }
}
```

```
// Setup Error Checking and reference the central error handler
private enum XmlErrorCheckingLevel {WELL_FORMED, VALID}

static private void setupAndCallSaxXmlReader(String fileName) throws Exception {

    final XmlErrorCheckingLevel ACTIVE_XML_ERROR_CHECKING_LEVEL =
            XmlErrorCheckingLevel.VALID;

    final String JAXP_SCHEMA_LANGUAGE = "http://java.sun" + "" +
            ".com/xml/jaxp/properties/schemaLanguage";
//        final String JAXP_SCHEMA_SOURCE = "http://java.sun" + "" +
//                ".com/xml/jaxp/properties/schemaSource";
    final String W3C_XML_SCHEMA = "http://www.w3.org/2001/XMLSchema";


    SAXParserFactory spf = SAXParserFactory.newInstance();
    spf.setNamespaceAware(true); // Required when using Validation.

    SAXParser saxParser;

    switch (ACTIVE_XML_ERROR_CHECKING_LEVEL) {
        case WELL_FORMED:
            spf.setValidating(false);
            saxParser = spf.newSAXParser();
            break;
        case VALID:
            spf.setValidating(true);
            saxParser = spf.newSAXParser();
            try {
                saxParser.setProperty(JAXP_SCHEMA_LANGUAGE, W3C_XML_SCHEMA);
            } catch (SAXNotRecognizedException x) {
                System.err.println(
                        "Error: JAXP SAXParser property not recognized: " +
                                JAXP_SCHEMA_LANGUAGE);

                System.err.println(
                        "Check to see if parser conforms to the JAXP spec.");
                System.exit(1);
            }

            break;
        default:
            throw new Exception("Unexpected Switch case reached");
    }

    XMLReader xmlReader = saxParser.getXMLReader();
    xmlReader.setContentHandler(new SaxLocalNameCount());
    xmlReader.setErrorHandler(new CentralSaxErrorHandler(System.err));
    xmlReader.parse(convertToPathSlashFileName(fileName));
}
```

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/jaxp/sax/parsing.html
(Oracle, 2017) http://docs.oracle.com/javase/tutorial/jaxp/sax/validation.html
C:\Users\John\Documents\Sda\Code\BjaxXsl\Examples\XmlAviationWeather\JaxpDemo\src\CentralSaxErrorHandler.java
C:\Users\John\Documents\Sda\Code\BjaxXsl\Examples\XmlAviationWeather\JaxpDemo\src\JaxpDemo.java

Refactor to add attribute and character (value node) handling.

```
public class SaxLocalNameCount extends org.xml.sax.helpers.DefaultHandler {

    private Map<String, Integer> mTags;
    private StringBuffer mStringBuffer;

    @Override
    public void startDocument() throws SAXException {
        super.startDocument();

        mTags = new LinkedHashMap<String, Integer>();
    }

    @Override
    public void startElement(String uri, String localName, String qName,
                             Attributes attributes) throws SAXException {
```

```
        super.startElement(uri, localName, qName, attributes);

        String key = localName;
        Object value = mTags.get(key);

        if (value == null) {
            mTags.put(key, new Integer(1));
        } else {
            int count = ((Integer) value).intValue();
            count++;
            mTags.put(key, new Integer(count));
        }

        System.out.println();

        // Output a line break between reports
        if (localName == "report") {
            System.out.println();
        }

        System.out.print(localName);

        for (int i = 0; i < attributes.getLength() ; i++) {
            System.out.printf(" [%s]", attributes.getValue(i));
        }
    }

    @Override
    public void characters(char[] chars, int start, int length) throws SAXException {
        super.characters(chars, start, length);
        mStringBuffer = new StringBuffer(length);
        // Be sure and only obtain the characters inside an element.
        mStringBuffer.append(Arrays.copyOfRange(chars, start, start + length));
    }

    @Override
    public void endElement(String uri, String localName,
                           String qName) throws SAXException {
        super.endElement(uri, localName, qName);
        if (mStringBuffer.length() > 0) {
            System.out.format(" : %s", mStringBuffer.toString());
            mStringBuffer.setLength(0); // Clear the buffer
        }
    }

    @Override
    public void endDocument() throws SAXException {
        super.endDocument();

        System.out.println();
        System.out.println("*** End of document report ***");
        for (Map.Entry<String, Integer> e : mTags.entrySet()) {
            System.out.format("Local Name '%s' occurs %d times.%n", e.getKey(),
                              e.getValue());
        }
    }
}
```

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/jaxp/sax/parsing.html ("Character events")
C:\Users\John\Documents\Sda\Code\BjaxXsl\Examples\XmlAviationWeather\JaxpDemo\src\SaxLocalNameCount.java

Results, with attribute and character handling.

```
report
location : YSSY
dateTimeUtc : 2003-10-23T15:30:00Z
wind
direction [magnetic] : 060
speed [knots] : 23
qnh [hectoPascals] : 1013.12
temperature [celsius] : 25
dewPoint [celsius] : -02
sky
```

```
cloud [few (0-2 eights)]
height [feet] [agl] : 4000
...

*** End of document report ***
Local Name 'takeOffAndLandingWeatherReports' occurs 1 times.
Local Name 'report' occurs 3 times.
Local Name 'location' occurs 3 times.
Local Name 'dateTimeUtc' occurs 3 times.
```

C:\Users\John\Documents\Sda\Code\BjaxXsl\Examples\XmlAviationWeather\JaxpDemo\src\SaxLocalNameCount.java

Sax Write XML (from arbitrary data source)

Setup and obtain a datasource, in this example a text file, and hand it off to do Sax processing.

```
public class JaxpDemo {

    /**
     * @param prefixOrSuffixString
     * @param args
     * @return the argument. Throw an error if not found
     * Examples:
     * <p/>
     * "-apiType" (prefix) returns "-apiType:sax" (if present)
     * ".xsd" (suffix) returns " takeOffAndLandingWeatherReports.xsd" (if present)
     */
    static private String getArgument(String prefixOrSuffixString, String args[]) {
        String result = "";
        for (String arg : args) {
            if (arg.matches(prefixOrSuffixString + ".*|.*\\" + prefixOrSuffixString)) {
                result = arg;
            }
        }
        if (result.equals("")) {
            System.err.printf("Argument with prefix or suffix '%s' not found%n",
                              prefixOrSuffixString);
            usage();
        }
        return result;
    }

    static public void main(String[] args) throws Exception {

        String xmlFileName = "";
        String xmlPathSlashFileName = "";

        String txtFileName = "";
        String xsltFileName = "";
        String xsdFileName = "";

        String apiType = getArgument("-apiType", args);
        String readType = getArgument("-readType", args);

        switch (apiType.toLowerCase()) {
            case "-apitype:sax":
                if (readType.toLowerCase().equals("-readtype:readxml")) {
....

                } else if ((readType.toLowerCase().equals("-readtype:writexml"))) {
                    //  Example arguments: -apiType:sax -readType:writeXml
                    // takeOffAndLandingWeatherReportValuePairs.txt

                    txtFileName = getArgument(".txt", args);
                    System.out.format("txtFileName:%s%n%n", txtFileName);
                    SaxMain.SaxWriteXmlFromArbitraryDataStructure(txtFileName,
SaxMain.OutputDirection.SCREEN);
                } else {
                    throw new UnexpectedException("Not expecting -readType: " + readType);
                }
```

```
                break;
            ....
            default:
                throw new UnexpectedException("apiType argument not supported. Must" +
                                                " be one of " +
                                                "'sax'|'dom'|'xsl'|'stax'. " +
                                                "Supplied: " + apiType);
        }
    }

    static private void usage() {
        System.err.println("Usage: JaxpDemo " +
                                "-apiType:['sax'|'dom'|'xsl'|'staxCursor'|'staxIterator]
" +
                                "-readType:[readXml|writeXml]  <file.xml>|<file" +
                                ".txt>|<file" + "" +
                                ".xml> <file.xslt>]");
        System.err.println("       -usage or -help: this message");
        System.err.println("       [But run from IntelliJ]");
        System.exit(1);
    }

}
```

C:\Users\John\Documents\Sda\Code\BjaxXsl\Examples\XmlAviationWeather\JaxpDemo\src\JaxpDemo.java

Implement a saxReader (from org.xml.sax.XMLReader) to parse your arbitrary data source.

```
public class WeatherReportTextFileReader implements XMLReader {

    ContentHandler mContentHandler = null;

    String mNamespaceUri = "";
    String mRootElement = "report";
    AttributesImpl mAttributes = new AttributesImpl();

    String mIndent = "\n  ";

    @Override
    public void parse(InputSource inputSource) throws IOException, SAXException {
        BufferedReader bufferedReader = new BufferedReader(
                inputSource.getCharacterStream());
        Scanner scannerInputSource = new Scanner(bufferedReader);
        try {

            if (mContentHandler == null) {
                throw new SAXException("No content handler");
            }

            // Insert newline
            mContentHandler.ignorableWhitespace("\n".toCharArray(), 0, 1);
            mContentHandler.startDocument();
            mContentHandler.startElement(mNamespaceUri, mRootElement, mRootElement,
                                         mAttributes);
            // Any number of spaces surrounding a tilde "~" or any number of spaces
            // till End cool of Line.
            String field = "";
            String value = "";

            while (scannerInputSource.hasNextLine()) {
                String line = scannerInputSource.nextLine();
                String[] lineTokens = line.split("~");
                if (lineTokens.length == 1) {
                    field = lineTokens[0];
                    value = "";
                } else if (lineTokens.length == 2) {
                    field = lineTokens[0];
                    value = lineTokens[1].trim();
                } else {
                    throw new UnexpectedException(
                            "lineTokens has unexpected length: " + lineTokens.length);
                }
```

```
                if (field.matches("qnh")) {
                    mAttributes.addAttribute(mNamespaceUri, "unit", "unit", "string",
                                             "hectoPascals");
                }
                xmlElementOutput(field, value, mAttributes);
                while (mAttributes.getLength() != 0) {
                    mAttributes.removeAttribute(0);
                }
            }

            mContentHandler.ignorableWhitespace("\n".toCharArray(), 0, 1);
            mContentHandler.endElement(mNamespaceUri, mRootElement, mRootElement);
            mContentHandler.endDocument();

        } catch (UnexpectedException e) {
            e.printStackTrace();
        }
    }

    private void xmlElementOutput(String elementLocalName, String elementTextNodeValue,
                                  AttributesImpl attributes) throws SAXException {
        mContentHandler.ignorableWhitespace(mIndent.toCharArray(), 0, mIndent.length());

        mContentHandler.startElement(mNamespaceUri, elementLocalName, elementLocalName
        /* qName */, attributes);
        mContentHandler.characters(elementTextNodeValue.toCharArray(), 0,
                                   elementTextNodeValue.length());
        mContentHandler.endElement(mNamespaceUri, elementLocalName, elementLocalName);
    }

    @Override
    public void setContentHandler(ContentHandler contentHandler) {
        this.mContentHandler = contentHandler;
    }

    @Override
    public ContentHandler getContentHandler() {
        return this.mContentHandler;
    }

    /**
     * Methods we need to implement, that we don't give a fuck about for the purposes
     * of this exercise.
     */

    @Override
    public void parse(String s) throws IOException, SAXException {
    }

    @Override
    public void setErrorHandler(ErrorHandler errorHandler) {
    }

    @Override
    public ErrorHandler getErrorHandler() {
        return null;
    }

    @Override
    public DTDHandler getDTDHandler() {
        return null;
    }

    @Override
    public EntityResolver getEntityResolver() {
        return null;
    }

    @Override
    public boolean getFeature(
            String s) throws SAXNotRecognizedException, SAXNotSupportedException {
        return false;
    }

    @Override
    public Object getProperty(
            String s) throws SAXNotRecognizedException, SAXNotSupportedException {
```

```
        return null;
    }


    @Override
    public void setDTDHandler(DTDHandler dtdHandler) {

    }

    @Override
    public void setEntityResolver(EntityResolver entityResolver) {

    }

    @Override
    public void setFeature(String s,
                           boolean b) throws SAXNotRecognizedException,
            SAXNotSupportedException {

    }

    @Override
    public void setProperty(String s,
                            Object o) throws SAXNotRecognizedException,
            SAXNotSupportedException {

    }
}
```

C:\Users\John\Documents\Sda\Code\BjaxXsl\Examples\XmlAviationWeather\JaxpDemo\src\sax\text\reader\WeatherReportTextFileReader.java

Do do following:

- Create a saxSource which joins a saxReader (as created above in step 2.) with an inputSource (in this case a file);
- Create a Stream result (in this case we output to the screen, but the code below illustrates the option to write to file);
- Use javax.xml.transform.Transformer to plug the saxSource into the Stream result.

```
public enum OutputDirection {
    FILE,
    SCREEN
}

static public void SaxWriteXmlFromArbitraryDataStructure(String txtFileName,
                                                         OutputDirection
                                                                 outputDirection)
        throws UnexpectedException {
    WeatherReportTextFileReader saxReader = new WeatherReportTextFileReader();
    File txtFile = new File(txtFileName);

    BufferedReader bufferedReader = null;
    try {
        bufferedReader = new BufferedReader(new FileReader(txtFile));

    } catch (FileNotFoundException e) {
        e.printStackTrace();
    }
    InputSource inputSource = new InputSource(bufferedReader);
    SAXSource saxSource = new SAXSource(saxReader, inputSource);

    StreamResult streamResult = null;
    switch (outputDirection) {
        case FILE:
            try {
                FileOutputStream fileOutputStream = new FileOutputStream(
                        "takeOffAndLandingWeatherReportResult.xml");
                streamResult = new StreamResult(fileOutputStream);
            } catch (FileNotFoundException e) {
                e.printStackTrace();
```

```
            }
            break;

        case SCREEN:
            streamResult = new StreamResult(System.out);
            break;

        default:
            throw new UnexpectedException("Unexpected outputDirection: " +
                                                outputDirection.toString());
    }

    TransformerFactory transformerFactory = TransformerFactory.newInstance();
    Transformer transformer = null;
    try {
        transformer = transformerFactory.newTransformer();
        transformer.setOutputProperty(OutputKeys.OMIT_XML_DECLARATION, "no");
        transformer.transform(saxSource, streamResult);
    } catch (TransformerConfigurationException e) {
        e.printStackTrace();
    } catch (TransformerException e) {
        e.printStackTrace();
    }
}
```

C:\Users\John\Documents\Sda\Code\BjaxXsl\Examples\XmlAviationWeather\JaxpDemo\src\sax\SaxMain.java

The input and output

```
// Input (takeOffAndLandingWeatherReportValuePairs.txt)
location~ YMML
dateTimeUtc~ 2003-05-16T16:00:00Z
wind~ 06014
direction~ 060
speed~ 14
qnh~ 1013.2
temperature~ 05
dewPoint~ 02
sky~
cavok~
remarks~ Runway 16L closed for repair.

// Output (To Screen)
txtFileName:takeOffAndLandingWeatherReportValuePairs.txt

<?xml version="1.0" encoding="UTF-8"?>
<report>
  <location>YMML</location>
  <dateTimeUtc>2003-05-16T16:00:00Z</dateTimeUtc>
  <wind>06014</wind>
  <direction>060</direction>
  <speed>14</speed>
  <qnh unit="hectoPascals">1013.2</qnh>
  <temperature>05</temperature>
  <dewPoint>02</dewPoint>
  <sky/>
  <cavok/>
  <remarks>Runway 16L closed for repair.</remarks>
</report>
Process finished with exit code 0
```

C:\Users\John\Documents\Sda\Code\BjaxXsl\Examples\XmlAviationWeather\XmlSource\takeOffAndLandingWeatherReportValuePairs.txt

Document Object Model (DOM)

Basics

See also (Bentley, EcmaScript in Web Quick Reference, 2004) EcmaScript in Web Quick Reference.docx#W3CDom .

In DOM the value of an element is not the same as its content. In particular the nodeValue() of an element is null. You'll need to fetch children text nodes to get an element's content.

```
// Example element in document
<sentence>This is an <bold>important</bold> idea.</sentence>

// DOM sees this structured as serveral nodes
ELEMENT: sentence
   + TEXT: This is an
   + ELEMENT: bold
     + TEXT: important
   + TEXT: idea.

// DOM methods return the following values for nodes
Node 1 (ELEMENT)
nodeName() = Sentence
nodeValue() = null

Node 2 (TEXT)
nodeName() = #text
nodeValue() = "This is an "
```

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/jaxp/dom/when.html

Table 3-1 Node Types, their names and their values.

Node	nodeName	nodeValue	Attributes
Attr	Name of attribute	Value of attribute	null
CDATASection	#cdata-section	Content of the CDATA section	null
Comment	#comment	Content of the comment	null
Document	#document	null	null
DocumentFragment	#documentFragment	null	null
DocumentType	Document Type name	null	null
Element	Tag name	null	null
Entity	Entity name	null	null
EntityReference	Name of entity referenced	null	null
Notation	Notation name	null	null
ProcessingInstruction	Target	Entire content excluding the target	null
Text	#text	Content of the text node	null

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/jaxp/dom/readingXML.html

Therefore when processing a DOM you must inspect the list of sub-nodes to "put together" the text of the node - even if that list contains only one item (a TEXT node).

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/jaxp/dom/when.html

DOM Coding Procedure

Set up and call parser. Include basic features: namespace awareness, XSD Scheme validation, lexical controls, calling a central error handler, do some work with the parsed result.

```
public class DomMain {

    static final String OUTPUT_ENCODING = "UTF-8";
    static final boolean VALIDATE = true;

    static final String JAXP_SCHEMA_LANGUAGE = "http://java.sun" + "" +
            ".com/xml/jaxp/properties/schemaLanguage";
    static final String W3C_XML_SCHEMA = "http://www.w3.org/2001/XMLSchema";

    public static void setupAndCallDomParser(
            String pathSlashFileName) throws ParserConfigurationException, IOException,
            SAXException {

        DocumentBuilderFactory dbf = DocumentBuilderFactory.newInstance();

        dbf.setNamespaceAware(true);

        // Validation
        dbf.setValidating(VALIDATE);
        if (VALIDATE) {
            try{
                dbf.setAttribute(JAXP_SCHEMA_LANGUAGE, W3C_XML_SCHEMA);
            } catch (IllegalArgumentException e) {
                System.err.println("Error: JAXP DocumentBuilderFactory attribute "
                                   + "not recognized: " + JAXP_SCHEMA_LANGUAGE);
                System.err.println("Check to see if parser conforms to JAXP spec.");
                System.exit(1);
            }
        }

        // Lexical Controls
        dbf.setIgnoringComments(false);
        dbf.setIgnoringElementContentWhitespace(false); // Only operates when validating.
        dbf.setCoalescing(false);
        dbf.setExpandEntityReferences(false);

        DocumentBuilder documentBuilder = dbf.newDocumentBuilder();

        // Call Error Handler
        OutputStreamWriter errorWriter = new OutputStreamWriter(System.err,
                                                                OUTPUT_ENCODING);
        documentBuilder.setErrorHandler(
                new CentralDomErrorHandler(new PrintWriter(errorWriter, true)));

        Document document = documentBuilder.parse(pathSlashFileName);

        // Do work with the DOM document
        DomEcho de = new DomEcho(document);
    }
}
```

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/jaxp/dom/readingXML.html
C:\Users\John\Documents\Sda\Code\BjaxXsl\Examples\XmlAviationWeather\JaxpDemo\src\dom\DomMain.java

Include a Central DOM Error Handler.

```
public class CentralDomErrorHandler implements ErrorHandler {
```

```
    private PrintWriter out;

    CentralDomErrorHandler(PrintWriter out) {
        this.out = out;
    }

    private String getParseExceptionInfo(SAXParseException spe) {
        String systemId = spe.getSystemId();
        if (systemId == null) {
            systemId = "null";
        }

        String info = "URI=" + systemId + " Line=" + spe.getLineNumber() +
                ": " + spe.getMessage();
        return info;
    }

    public void warning(SAXParseException spe) throws SAXException {
        out.println("Warning: " + getParseExceptionInfo(spe));
    }

    public void error(SAXParseException spe) throws SAXException {
        String message = "Error: " + getParseExceptionInfo(spe);
        throw new SAXException(message);
    }

    public void fatalError(SAXParseException spe) throws SAXException {
        String message = "Fatal Error: " + getParseExceptionInfo(spe);
        throw new SAXException(message);
    }
}
```

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/jaxp/dom/readingXML.html
C:\Users\John\Documents\Sda\Code\BjaxXsl\Examples\XmlAviationWeather\JaxpDemo\src\dom\CentralDomErrorHandler.java

Traverse the DOM in some way, Tree Walker recommended by default (it works best with a custom filter).

```
public class DomEcho {

    private PrintWriter out;
    private Document mDocument = null;


    DomEcho(Document document) {
        this.mDocument = document;

        out = new PrintWriter(System.out);
        traverseNodesUsingTreeWalker(document);
        out.flush();
    }

    class MyNodeFilter implements NodeFilter {
        @Override
        public short acceptNode(Node node) {
            if (node.getNodeType() == Node.ELEMENT_NODE) {
                if (node.getLocalName().equals("wind")) {
                    return FILTER_REJECT;
                }
            }
            return FILTER_ACCEPT;
        }
    }


    private void traverseNodesUsingTreeWalker(Document document) {
        // whatToShow flags filter out nodes before custom filter. If a whatToShow flag
        // effects a filter then the custom filter is not called.
        TreeWalker tw = ((DocumentTraversal) document).createTreeWalker(document,
                                                                        WHAT_TO_SHOW,
                                                                        new MyNodeFilter(),
                                                                        true);
```

```
        processTreeWalker(tw);
    }

    private void processTreeWalker(TreeWalker tw) {
        Node currentNode = tw.getCurrentNode();
        printlnNodeInfoSimple(currentNode);
        for (Node childNode = tw.firstChild(); childNode != null; childNode = tw
                .nextSibling()) {
            processTreeWalker(tw);
        }
        tw.setCurrentNode(currentNode);
    }
```

(Orcale, 2007) http://docs.oracle.com/cd/B28359_01/appdev.111/b28391/org/w3c/dom/traversal/TreeWalker.html
C:\Users\John\Documents\Sda\Code\BjaxXsl\Examples\XmlAviationWeather\JaxpDemo\src\dom\DomEcho.java

Display node information. Displaying node information generally entails sharing some responsibility to finish traversing some of the nodes. Chiefly, we need to recurse the node information function to traverse the attribute nodes for a given element node.

```
private void printlnNodeInfoSimple(Node node) {
    short nodeType = node.getNodeType();

    switch (nodeType) {
        case Node.ATTRIBUTE_NODE:
            out.printf(" [%s=\"%s\"] ", node.getLocalName(), node.getNodeValue());
            break;

        case Node.ELEMENT_NODE:
            // Custom element handling: insert line break in output
            if (node.getLocalName() == "report") {
                out.println();
            }
            out.printf("%n%s ", node.getLocalName());

            if ((WHAT_TO_SHOW & NodeFilter.SHOW_ATTRIBUTE) == NodeFilter
                    .SHOW_ATTRIBUTE) {

                NamedNodeMap attributes = node.getAttributes();
                mIndentLevel += 2;
                for (int i = 0; i < attributes.getLength(); i++) {
                    Node attribute = attributes.item(i);
                    printlnNodeInfoSimple(attribute);
                }
                mIndentLevel -= 2;
            }
            break;

        case Node.TEXT_NODE:
            if (node.getNodeValue().trim().equals("")) {
                // ignore white space node
            } else {
                out.printf("{%s}", node.getNodeValue());
            }
            break;

        default:
            // Ignore other nodes
    }
}
```

C:\Users\John\Documents\Sda\Code\BjaxXsl\Examples\XmlAviationWeather\JaxpDemo\src\dom\DomEcho.java

Optionally filter nodes using the whatToShow filter and/or a custom filter.

```
public class DomEcho {
    private static final String BASIC_INDENT = "  ";
    private static final int WHAT_TO_SHOW = NodeFilter.SHOW_ELEMENT | NodeFilter
            .SHOW_ATTRIBUTE | NodeFilter.SHOW_TEXT;
    //        private static final int WHAT_TO_SHOW = NodeFilter.SHOW_ALL;
```

```
    private int mIndentLevel = 0;

    private PrintWriter out;
    private Document mDocument = null;


    DomEcho(Document document) {
        this.mDocument = document;

        out = new PrintWriter(System.out);
        //        traverseNodesUsingNodeIterator(document);
        traverseNodesUsingTreeWalker(document);
        //        printlnNodeInfoTypeDependent(document);
        out.flush();
    }

    class MyNodeFilter implements NodeFilter {
        @Override
        public short acceptNode(Node node) {
            if (node.getNodeType() == Node.ELEMENT_NODE) {
                if (node.getLocalName().equals("wind")) {
                    return FILTER_REJECT;
                }
            }
            return FILTER_ACCEPT;
        }
    }

    private void traverseNodesUsingTreeWalker(Document document) {
        // whatToShow flags filter out nodes before custom filter. If a whatToShow flag
        // effects a filter then the custom filter is not called.
        TreeWalker tw = ((DocumentTraversal) document).createTreeWalker(document,
                                                                        WHAT_TO_SHOW,
                                                                        new MyNodeFilter(),
                                                                        true);
        processTreeWalker(tw);
    }
```

(W3C, 2000) *http://www.w3.org/TR/2000/REC-DOM-Level-2-Traversal-Range-20001113/traversal.html#Traversal-Filters*
(Orcale, 2007) *http://docs.oracle.com/cd/B28359_01/appdev.111/b28391/org/w3c/dom/traversal/NodeIterator.html*
(Orcale, 2007) *http://docs.oracle.com/cd/B28359_01/appdev.111/b28391/org/w3c/dom/traversal/TreeWalker.html*
C:\Users\John\Documents\Sda\Code\BjaxXsl\Examples\XmlAviationWeather\JaxpDemo\src\dom\DomEcho.java

XML Schema Validation

XML validation requires:

- At least one schema associated with the xml document.
- Setting the document builder factory to validate.
- Setting the document builder factory with Java and Schema url constants.
- An error handler that has been set;

```
public class DomMain {

    static final String OUTPUT_ENCODING = "UTF-8";
    static final boolean VALIDATE = true;

    static final String JAXP_SCHEMA_LANGUAGE = "http://java.sun" + "" +
            ".com/xml/jaxp/properties/schemaLanguage";
    static final String W3C_XML_SCHEMA = "http://www.w3.org/2001/XMLSchema";

    public static void setupAndCallDomParser(
            String pathSlashFileName) throws ParserConfigurationException, IOException,
            SAXException {

        DocumentBuilderFactory dbf = DocumentBuilderFactory.newInstance();
```

```
        dbf.setNamespaceAware(true);

        // In this example the XML Schema is associated in XML document

        // Validation
        dbf.setValidating(VALIDATE);
        if (VALIDATE) {
            try{
                dbf.setAttribute(JAXP_SCHEMA_LANGUAGE, W3C_XML_SCHEMA);
            } catch (IllegalArgumentException e) {
                System.err.println("Error: JAXP DocumentBuilderFactory attribute "
                                   + "not recognized: " + JAXP_SCHEMA_LANGUAGE);
                System.err.println("Check to see if parser conforms to JAXP spec.");
                System.exit(1);
            }
        }

        // Lexical Controls
        dbf.setIgnoringComments(false);
        dbf.setIgnoringElementContentWhitespace(true); // Only operates when validating.
        dbf.setCoalescing(false);
        dbf.setExpandEntityReferences(false);

        DocumentBuilder documentBuilder = dbf.newDocumentBuilder();

        // Set Error Handler
        OutputStreamWriter errorWriter = new OutputStreamWriter(System.err,
                                                                OUTPUT_ENCODING);
        documentBuilder.setErrorHandler(
                new CentralDomErrorHandler(new PrintWriter(errorWriter, true)));

        Document document = documentBuilder.parse(pathSlashFileName);

        // Do work with the DOM document
        DomEcho de = new DomEcho(document);
    }
}
```

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/jaxp/dom/validating.html

You can associate and XSD Schema with an xml document in two ways:

In the XML document itself; or

```
<takeOffAndLandingWeatherReports
xmlns="http://tempuri.org/TakeOffAndLandingWeatherReports.xsd"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://tempuri.org/TakeOffAndLandingWeatherReports.xsd
TakeOffAndLandingWeatherReports.xsd">
```

C:\Users\John\Documents\Sda\Code\BjaxXsl\Examples\XmlAviationWeather\XmlSource\takeOffAndLandingWeatherReports.xml

In Java Code.

```
public class DomMain {
...

    static final String JAXP_SCHEMA_SOURCE = "http://java.sun" +
            ".com/xml/jaxp/properties/schemaSource";

    public static void setupAndCallDomParser(
            String pathSlashFileName) throws ParserConfigurationException, IOException,
            SAXException {

        String schemaSource =
                "takeOffAndLandingWeatherReports.xsd";

        DocumentBuilderFactory dbf = DocumentBuilderFactory.newInstance();

        dbf.setNamespaceAware(true);
```

```
        // Validation
        dbf.setValidating(VALIDATE);
        if (VALIDATE) {
            try {
                dbf.setAttribute(JAXP_SCHEMA_LANGUAGE, W3C_XML_SCHEMA);
            } catch (IllegalArgumentException e) {
                System.err.println(
                        "Error: JAXP DocumentBuilderFactory attribute " + "not " +
                                "recognized: " + JAXP_SCHEMA_LANGUAGE);
                System.err.println("Check to see if parser conforms to JAXP spec.");
                System.exit(1);
            }

            if (schemaSource != null) {
                dbf.setAttribute(JAXP_SCHEMA_SOURCE, new File(schemaSource));
            }
        }
```

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/jaxp/dom/validating.html
C:\Users\John\Documents\Sda\Code\BjaxXsl\Examples\XmlAviationWeather\JaxpDemo\src\dom\DomMain.java

For validating with multiple namespaces see ...

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/jaxp/dom/validating.html

DOM Traverse and Display

See also (Bentley, EcmaScript in Web Quick Reference, 2004) EcmaScript in Web Quick Reference > W3C Dom

Traverse the DOM

There are several ways to traverse the DOM:

Direct recursion from node. Use node.getNextSibling and other node lists. Other node methods include: getFirstChild, getLastChild, getPreviousSibling, and getParentNode.

```
// W3C Direct. Recurse with node pattern.
processMe(Node n) {
   nodeStartActions(n);
   for (Node child=n.firstChild();
        child != null;
        child=child.nextSibling()) {
      processMe(child);
   }
   nodeEndActions(n);
}
```

(W3C, 2000) http://www.w3.org/TR/2000/REC-DOM-Level-2-Traversal-Range-20001113/traversal.html, "TreeWalker" section.

```
// Oracle Java Documentation Direct. Recurse with node pattern.
rivate void printlnNodeInfoTypeDependent(Node node) {
    outputIndentation();

    short nodeType = node.getNodeType();
    switch(nodeType){
        case Node.ATTRIBUTE_NODE:
            out.print("ATTRIBUTE_NODE:");
            printlnNodeInfoCommon(node);
            break;
..

        case Node.DOCUMENT_TYPE_NODE:
```

```
            out.print("DOCUMENT_TYPE_NODE:");
            printlnNodeInfoCommon(node);
            NamedNodeMap namedNodeMap = ((DocumentType)node).getEntities();
            mIndentLevel += 2;
            for (int i = 0; i < namedNodeMap.getLength(); i++) {
                Entity entity = (Entity) namedNodeMap.item(i);
                printlnNodeInfoTypeDependent(entity);
            } (W3C, 2000)
            mIndentLevel -= 2;
            break;

        case Node.ELEMENT_NODE:
            // Custom element handling: insert line break in output
            if (node.getLocalName() == "report") {
                out.println();
            }
            out.print("ELEMENT_NODE");
            printlnNodeInfoCommon(node);

            NamedNodeMap attributes = node.getAttributes();
            mIndentLevel += 2;
            for (int i = 0; i < attributes.getLength(); i++) {
                Node attribute = attributes.item(i);
                printlnNodeInfoTypeDependent(attribute);
            }
            mIndentLevel -= 2;
            break;

        case Node.ENTITY_NODE:
            out.print("ENTITY_NODE:");
            printlnNodeInfoCommon(node);
            break;

        case Node.ENTITY_REFERENCE_NODE:
            out.print("ENTITY_REFERENCE_NODE:");
            printlnNodeInfoCommon(node);
            break;

        case Node.NOTATION_NODE:
            out.print("NOTATION_NODE:");
            printlnNodeInfoCommon(node);
            break;

        case Node.PROCESSING_INSTRUCTION_NODE:
            out.print("PROCESSING_INSTRUCTION_NODE:");
            printlnNodeInfoCommon(node);
            break;

        case Node.TEXT_NODE:
            out.print("TEXT_NODE:");
            printlnNodeInfoCommon(node);
            break;

        default:
            out.print("UNSUPPORTED NODE TYPE: " + nodeType);
    }

    mIndentLevel++;
    for (Node childNode = node.getFirstChild(); childNode != null; childNode =
            childNode.getNextSibling()) {
        printlnNodeInfoTypeDependent(childNode);
    }
    mIndentLevel--;
}
```

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/jaxp/dom/readingXML.html
C:\Users\John\Documents\Sda\Code\BjaxXsl\Examples\XmlAviationWeather\JaxpDemo\src\dom\DomEcho.java

Node iterator.

```
public class DomEcho {
    private static final String OUTPUT_ENCODING = "UTF-8";
    private static final String BASIC_INDENT = "  ";
    private static final int WHAT_TO_SHOW = NodeFilter.SHOW_ELEMENT | NodeFilter
```

```
            .SHOW_ATTRIBUTE | NodeFilter.SHOW_TEXT;
//        private static final int WHAT_TO_SHOW = NodeFilter.SHOW_ALL;

...


    DomEcho(Document document) {
        this.mDocument = document;

        out = new PrintWriter(System.out);
        traverseNodesUsingNodeIterator(document);
        out.flush();
    }


    private void traverseNodesUsingNodeIterator(Document document) {

        // whatToShow flags filter out nodes before custom filter. If a whatToShow flag
        // effects a filter then the custom filter is not called.
        NodeIterator ni = ((DocumentTraversal) document).createNodeIterator(document,
                                                                            WHAT_TO_SHOW,
                                                                            null
                                                                            true);

        for (Node node = ni.nextNode(); node != null; node = ni.nextNode()) {
            printlnNodeInfoSimple(node);
        }
    }
```

(Orcale, 2007) *http://docs.oracle.com/cd/B28359_01/appdev.111/b28391/org/w3c/dom/traversal/NodeIterator.html*
(W3C, 2000) *http://www.w3.org/TR/2000/REC-DOM-Level-2-Traversal-Range-20001113/traversal.html#Iterator-overview*
C:\Users\John\Documents\Sda\Code\BjaxXsl\Examples\XmlAviationWeather\JaxpDemo\src\dom\DomEcho.java

Tree walker recursion (recommended).

```
// What I think the W3C intends
processMe(TreeWalker tw) {
   Node n = tw.getCurrentNode();
   nodeStartActions(n);
   for (Node child=tw.firstChild();
        child!=null;
        child=tw.nextSibling()) {
      processMe(tw);
   }

   tw.setCurrentNode(n);
   nodeEndActions(n);
}

// As it is in W3C documenation
processMe(TreeWalker tw) {
   Node n = tw.getCurrentNode();
   nodeStartActions(tw);
   for (Node child=tw.firstChild();
        child!=null;
        child=tw.nextSibling()) {
      processMe(tw);
   }

   tw.setCurrentNode(n);
   nodeEndActions(tw);
}
```

(W3C, 2000) *http://www.w3.org/TR/2000/REC-DOM-Level-2-Traversal-Range-20001113/traversal.html#TreeWalker*

```
public class DomEcho {
    private static final String OUTPUT_ENCODING = "UTF-8";
    private static final String BASIC_INDENT = "  ";
    private static final int WHAT_TO_SHOW = NodeFilter.SHOW_ELEMENT | NodeFilter
            .SHOW_ATTRIBUTE | NodeFilter.SHOW_TEXT;
```

```
.....


    DomEcho(Document document) {
        this.mDocument = document;

        out = new PrintWriter(System.out);
        traverseNodesUsingTreeWalker(document);
        out.flush();
    }


    private void traverseNodesUsingTreeWalker(Document document) {
        // whatToShow flags filter out nodes before custom filter. If a whatToShow flag
        // effects a filter then the custom filter is not called.
        TreeWalker tw = ((DocumentTraversal) document).createTreeWalker(document,
                                                                        WHAT_TO_SHOW,
                                                                        new MyNodeFilter(),
                                                                        true);
        processTreeWalker(tw);
    }

    private void processTreeWalker(TreeWalker tw) {
        Node currentNode = tw.getCurrentNode();
        printlnNodeInfoSimple(currentNode); // Some kind of processing of the node.
        for (Node childNode = tw.firstChild(); childNode != null; childNode = tw
                .nextSibling()) {
            processTreeWalker(tw);
        }
        tw.setCurrentNode(currentNode);
    }
```

(Orcale, 2007) http://docs.oracle.com/cd/B28359_01/appdev.111/b28391/org/w3c/dom/traversal/TreeWalker.html
C:\Users\John\Documents\Sda\Code\BjaxXsl\Examples\XmlAviationWeather\JaxpDemo\src\dom\DomEcho.java

Displaying Node Information

Displaying node information generally entails sharing some responsibility to finish traversing some of the nodes. Chiefly, we need to recurse the node information function to traverse the attribute nodes for a given element node.

```
private void printlnNodeInfoSimple(Node node) {
    short nodeType = node.getNodeType();

    switch (nodeType) {
        case Node.ATTRIBUTE_NODE:
            out.printf(" [%s=\"%s\"] ", node.getLocalName(), node.getNodeValue());
            break;

        case Node.ELEMENT_NODE:
            // Custom element handling: insert line break in output
            if (node.getLocalName() == "report") {
                out.println();
            }
            out.printf("%n%s ", node.getLocalName());

            if ((WHAT_TO_SHOW & NodeFilter.SHOW_ATTRIBUTE) == NodeFilter
                    .SHOW_ATTRIBUTE) {

                NamedNodeMap attributes = node.getAttributes();
                mIndentLevel += 2;
                for (int i = 0; i < attributes.getLength(); i++) {
                    Node attribute = attributes.item(i);
                    printlnNodeInfoSimple(attribute);
                }
                mIndentLevel -= 2;
            }
            break;

        case Node.TEXT_NODE:
            if (node.getNodeValue().trim().equals("")) {
                // ignore white space node
```

```
            } else {
                out.printf("{%s}", node.getNodeValue());
            }
            break;

        default:
            // Ignore other nodes
    }
}
```

C:\Users\John\Documents\Sda\Code\BjaxXsl\Examples\XmlAviationWeather\JaxpDemo\src\dom\DomEcho.java

To display node information you can use some combination of the following methods (depending on your traversal method).

Good with Node iterator or Tree Walker.

```
private void printlnNodeInfoSimple(Node node) {
    short nodeType = node.getNodeType();

    switch (nodeType) {
        case Node.ATTRIBUTE_NODE:
            out.printf(" [%s=\"%s\"] ", node.getLocalName(), node.getNodeValue());
            break;

        case Node.ELEMENT_NODE:
            // Custom element handling: insert line break in output
            if (node.getLocalName() == "report") {
                out.println();
            }
            out.printf("%n%s ", node.getLocalName());

            if ((WHAT_TO_SHOW & NodeFilter.SHOW_ATTRIBUTE) == NodeFilter
                    .SHOW_ATTRIBUTE) {

                NamedNodeMap attributes = node.getAttributes();
                mIndentLevel += 2;
                for (int i = 0; i < attributes.getLength(); i++) {
                    Node attribute = attributes.item(i);
                    printlnNodeInfoSimple(attribute);
                }
                mIndentLevel -= 2;
            }
            break;

        case Node.TEXT_NODE:
            if (node.getNodeValue().trim().equals("")) {
                // ignore white space node
            } else {
                out.printf("{%s}", node.getNodeValue());
            }
            break;

        default:
            // Ignore other nodes
    }
}
```

C:\Users\John\Documents\Sda\Code\BjaxXsl\Examples\XmlAviationWeather\JaxpDemo\src\dom\DomEcho.java

Good with direct node recursion.

```
private void outputIndentation() {
    for (int i = 0; i < mIndentLevel; i++) {
        out.print(BASIC_INDENT);
    }
}

private void printlnNodeInfoCommon(Node node) {
    out.printf("  nodeName='%s'", node.getNodeName());
```

```
    String str = node.getNamespaceURI();
    if (str != null) {
        out.printf("  uri='%s'", str);
    }

    str = node.getPrefix();
    if (str != null) {
        out.printf("  pre='%s'", str);
    }

    str = node.getLocalName();
    if (str != null) {
        out.printf("  local='%s'", str);
    }

    str = node.getNodeValue();
    if (str != null) {
        out.print("  nodeValue=");
        if (str.trim().equals("")) {
            // Whitespace
            out.print("[Whitespace]");
        } else {
            out.printf("'%s'", str);
        }
    }
    out.println();
}

private void printlnNodeInfoTypeDependent(Node node) {
    outputIndentation();

    short nodeType = node.getNodeType();
    switch (nodeType) {
        case Node.ATTRIBUTE_NODE:
            out.print("ATTRIBUTE_NODE:");
            printlnNodeInfoCommon(node);
            break;

        case Node.CDATA_SECTION_NODE:
            out.print("CDATA_SECTION_NODE:");
            printlnNodeInfoCommon(node);
            break;

        case Node.COMMENT_NODE:
            out.print("COMMENT_NODE:");
            printlnNodeInfoCommon(node);
            break;

        case Node.DOCUMENT_NODE:
            out.print("DOCUMENT_NODE:");
            printlnNodeInfoCommon(node);
            break;

        case Node.DOCUMENT_FRAGMENT_NODE:
            out.print("DOCUMENT_FRAGMENT_NODE:");
            printlnNodeInfoCommon(node);
            break;

        case Node.DOCUMENT_TYPE_NODE:
            out.print("DOCUMENT_TYPE_NODE:");
            printlnNodeInfoCommon(node);
            NamedNodeMap namedNodeMap = ((DocumentType) node).getEntities();
            mIndentLevel += 2;
            for (int i = 0; i < namedNodeMap.getLength(); i++) {
                Entity entity = (Entity) namedNodeMap.item(i);
                printlnNodeInfoTypeDependent(entity);
            }
            mIndentLevel -= 2;
            break;

        case Node.ELEMENT_NODE:
            // Custom element handling: insert line break in output
            if (node.getLocalName() == "report") {
                out.println();
            }
            out.print("ELEMENT_NODE");
            printlnNodeInfoCommon(node);
```

```
            NamedNodeMap attributes = node.getAttributes();
            mIndentLevel += 2;
            for (int i = 0; i < attributes.getLength(); i++) {
                Node attribute = attributes.item(i);
                printlnNodeInfoTypeDependent(attribute);
            }
            mIndentLevel -= 2;
            break;

        case Node.ENTITY_NODE:
            out.print("ENTITY_NODE:");
            printlnNodeInfoCommon(node);
            break;

        case Node.ENTITY_REFERENCE_NODE:
            out.print("ENTITY_REFERENCE_NODE:");
            printlnNodeInfoCommon(node);
            break;

        case Node.NOTATION_NODE:
            out.print("NOTATION_NODE:");
            printlnNodeInfoCommon(node);
            break;

        case Node.PROCESSING_INSTRUCTION_NODE:
            out.print("PROCESSING_INSTRUCTION_NODE:");
            printlnNodeInfoCommon(node);
            break;

        case Node.TEXT_NODE:
            out.print("TEXT NODE:");
            printlnNodeInfoCommon(node);
            break;

        default:
            out.print("UNSUPPORTED NODE TYPE: " + nodeType);
    }

    mIndentLevel++;
    for (Node childNode = node.getFirstChild(); childNode != null; childNode =
            childNode.getNextSibling()) {
        printlnNodeInfoTypeDependent(childNode);
    }
    mIndentLevel--;
}
```

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/jaxp/dom/readingXML.html
C:\Users\John\Documents\Sda\Code\BjaxXsl\Examples\XmlAviationWeather\JaxpDemo\src\dom\DomEcho.java

Text utility function might come in handy

```
/**
 * Return the text that a node contains. This routine:
 * <ul>
 * <li>Ignores comments and processing instructions.
 * <li>Concatenates TEXT nodes, CDATA nodes, and the results of
 * recursively processing EntityRef nodes.
 * <li>Ignores any element nodes in the sublist.
 * (Other possible options are to recurse into element
 * sublists or throw an exception.)
 * </ul>
 *
 * @param node a  DOM node
 * @return a String representing its contents
 */
public String getText(Node node, boolean removeWhitespaceNodes) {
    StringBuffer result = new StringBuffer();
    if (!node.hasChildNodes()) return "";

    NodeList nodeList = node.getChildNodes();
    for (int i = 0; i < nodeList.getLength(); i++) {
        Node subNode = nodeList.item(i);
        if (subNode.getNodeType() == Node.TEXT_NODE) {
```

```
            if (removeWhitespaceNodes && subNode.getNodeValue().trim().equals("")) {
                // ignore white space node
            } else {
                result.append(subNode.getNodeValue());
            }

        } else if (subNode.getNodeType() == Node.CDATA_SECTION_NODE) {
            result.append(subNode.getNodeValue());
        } else if (subNode.getNodeType() == Node.ENTITY_REFERENCE_NODE) {
            // Recurse into the subtree for text
            // (and ignore comments)
            result.append(getText(subNode, removeWhitespaceNodes));
        }
    }

    return result.toString();
}
```

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/jaxp/dom/readingXML.html
C:\Users\John\Documents\Sda\Code\BjaxXsl\Examples\XmlAviationWeather\JaxpDemo\src\dom\DomEcho.java

Filtering Nodes during DOM Traversal

You can filter out the nodes to traverse in three ways:

Setting lexical controls before the parser is invoked.

```
    public static void setupAndCallDomParser(
            String pathSlashFileName) throws ParserConfigurationException, IOException,
            SAXException {

        DocumentBuilderFactory dbf = DocumentBuilderFactory.newInstance();

        dbf.setNamespaceAware(true);

        // Validation
        dbf.setValidating(VALIDATE);
        if (VALIDATE) {
            try{
                dbf.setAttribute(JAXP_SCHEMA_LANGUAGE, W3C_XML_SCHEMA);
            } catch (IllegalArgumentException e) {
                System.err.println("Error: JAXP DocumentBuilderFactory attribute "
                                   + "not recognized: " + JAXP_SCHEMA_LANGUAGE);
                System.err.println("Check to see if parser conforms to JAXP spec.");
                System.exit(1);
            }
        }

        // Lexical Controls
        dbf.setIgnoringComments(false);
        dbf.setIgnoringElementContentWhitespace(true); // Only operates when validating.
        dbf.setCoalescing(false);
        dbf.setExpandEntityReferences(false);

        DocumentBuilder documentBuilder = dbf.newDocumentBuilder();

        // Call Error Handler
        OutputStreamWriter errorWriter = new OutputStreamWriter(System.err,
                                                                OUTPUT_ENCODING);
        documentBuilder.setErrorHandler(
                new CentralDomErrorHandler(new PrintWriter(errorWriter, true)));

        Document document = documentBuilder.parse(pathSlashFileName);

        // Do work with the DOM document
        DomEcho de = new DomEcho(document);
    }
```

Using the whatToShow bitwise flag when using a NodeIterator or TreeWalker; and
Using a custom node filter.

```
public class DomEcho {
    private static final String OUTPUT_ENCODING = "UTF-8";
    private static final String BASIC_INDENT = "  ";
    private static final int WHAT_TO_SHOW = NodeFilter.SHOW_ELEMENT | NodeFilter
            .SHOW_ATTRIBUTE | NodeFilter.SHOW_TEXT;
//        private static final int WHAT_TO_SHOW = NodeFilter.SHOW_ALL;

    private int mIndentLevel = 0;

    private PrintWriter out;
    private Document mDocument = null;


    DomEcho(Document document) {
        this.mDocument = document;

        out = new PrintWriter(System.out);
        traverseNodesUsingTreeWalker(document);
        out.flush();
    }

    class MyNodeFilter implements NodeFilter {
        @Override
        public short acceptNode(Node node) {
            if (node.getNodeType() == Node.ELEMENT_NODE) {
                if (node.getLocalName().equals("wind")) {
                    return FILTER_REJECT;
                }
            }
            return FILTER_ACCEPT;
        }
    }

    private void traverseNodesUsingTreeWalker(Document document) {
        // whatToShow flags filter out nodes before custom filter. If a whatToShow flag
        // effects a filter then the custom filter is not called.
        TreeWalker tw = ((DocumentTraversal) document).createTreeWalker(document,
                                                                        WHAT_TO_SHOW,
                                                                        new MyNodeFilter(),
                                                                        true);

        processTreeWalker(tw);
    }
```

(W3C, 2000) http://www.w3.org/TR/2000/REC-DOM-Level-2-Traversal-Range-20001113/traversal.html#Traversal-Filters
(Orcale, 2007) http://docs.oracle.com/cd/B28359_01/appdev.111/b28391/org/w3c/dom/traversal/NodeIterator.html
(Orcale, 2007) http://docs.oracle.com/cd/B28359_01/appdev.111/b28391/org/w3c/dom/traversal/TreeWalker.html
C:\Users\John\Documents\Sda\Code\BjaxXsl\Examples\XmlAviationWeather\JaxpDemo\src\dom\DomEcho.java

If acceptNode() returns:

- FILTER_ACCEPT, the Node will be present in the logical view;
- FILTER_SKIP, the Node will not be present in the logical view, but the children of the Node may;
- FILTER_REJECT, neither the Node nor its descendants will be present in the logical view.
- Since iterators present nodes as an ordered list, without hierarchy, FILTER_REJECT and FILTER_SKIP are synonyms for NodeIterators, skipping only the single current node. That is, FILTER_REJECT only works in a Tree Walker to cut out a node and all it's descendents.

(W3C, 2000) http://www.w3.org/TR/2000/REC-DOM-Level-2-Traversal-Range-20001113/traversal.html#Traversal-Filters

Searching for Nodes

Search for nodes. Iterate over elements, and for each element search their subnodes for the target.

```
public class DomEcho {
    private static final String BASIC_INDENT = "  ";
    private static final int WHAT_TO_SHOW = NodeFilter.SHOW_ELEMENT;

    private int mIndentLevel = 0;

    private PrintWriter out;
    private Document mDocument = null;


    DomEcho(Document document) {
        this.mDocument = document;

        out = new PrintWriter(System.out);
        traverseNodesUsingNodeIterator(document);
        out.flush();
    }


    private void outputSubNodeValue(String subNodeName, Node nodeToSearch) {
        Node foundNode = searchForSubNodeValue(subNodeName, nodeToSearch);
        if (foundNode != null) {
            out.printf("Found subnode: %s with value %s%n", foundNode.getLocalName(),
                       getText(foundNode, true));
        }
    }

    private Node searchForSubNodeValue(String subNodeName, Node nodeToSearch) {
        if (nodeToSearch.getNodeType() != Node.ELEMENT_NODE) {
            System.err.println("Error: Search node not of element type");
            System.exit(22);
        }

        if (!nodeToSearch.hasChildNodes()) return null;
        NodeList nodeList = nodeToSearch.getChildNodes();
        for (int i = 0; i < nodeList.getLength(); i++) {
            Node childNode = nodeList.item(i);
            if (childNode.getNodeType() == Node.ELEMENT_NODE) {
                if (childNode.getLocalName().equals(subNodeName)) {
                    return childNode;
                }
            }
        }
        return  null;
    }

    private void traverseNodesUsingNodeIterator(Document document) {

        // whatToShow flags filter out nodes before custom filter. If a whatToShow flag
        // effects a filter then the custom filter is not called.
        NodeIterator ni = ((DocumentTraversal) document).createNodeIterator(document,
                                                                            WHAT_TO_SHOW,
                                                                            null,
                                                                            true);

        for (Node node = ni.nextNode(); node != null; node = ni.nextNode()) {
            outputSubNodeValue("location", node);
        }
    }
```

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/jaxp/dom/readingXML.html
C:\Users\John\Documents\Sda\Code\BjaxXsl\Examples\XmlAviationWeather\JaxpDemo\src\dom\DomEcho.java

DOM Manipulaton

For DOM manipulation algorithms see (Bentley, EcmaScript in Web Quick Reference, 2004) EcmaScript in Web Quick Reference.docx#W3CDom

Manipulate the DOM as follows.

Operation	Class	Example Methods	More Info
Create Node	Document interface	createElement, createComment, createCDATAsection, createTextNode	http://docs.oracle.com/javase/7/docs/api/org/w3c/dom/Document.html
Insert Nodes			Remember creating an element node only gives you a "hook to hang things on". You'll generally have to create the element's text node for example
Create Attributes	Element interface	setAttribute	http://docs.oracle.com/javase/7/docs/api/org/w3c/dom/Element.html
	Document interface	createAttribute & setAttributeNode	http://docs.oracle.com/javase/7/docs/api/org/w3c/dom/Document.html
Remove or Replace Node	Node interface	removeChild replaceChild setNodeValue	http://docs.oracle.com/javase/7/docs/api/org/w3c/dom/Node.html

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/jaxp/dom/readingXML.html

XSL (XSLT and XPath)

For XSLT and XPath themselves see XSL Quick Reference.doc and C:\Users\John\Documents\Sda\Code\BjaxXsl\Apps\XpathTester.

XSLT Coding Procedure

Coding procedure for performing an XSLT transformation in Java.

Obtain XML and XSLT files and pass them along.

```
/**
 * Call this from IntelliJ Run Configuration with a working directory set to
 * C:\Users\John\Documents\Sda\Code\BjaxXsl\Examples\XmlAviationWeather\XmlSource
 */
public class JaxpDemo {
    static public void main(String[] args) throws Exception {
        String argument = null;
        String xmlFileName = "";
        String xmlPathSlashFileName = "";
        String xsltFileName = "";
        String xsltPathSlashFileName = "";

        // Todo: refactor to handle two arguments .xml and .xslt
        // Display usage if there is more than one arg
        if (args.length != 2) {
            System.out.println("There must be two arguments");
            usage();
        } else {

            for (int i = 0; i < args.length; i++) {
                argument = args[i];
                // Display usage for any argument intended as a switch '-argument',
                if ((argument == null) || (argument.matches("-.*"))) {
```

```
                    System.out.println("You can't have a switch '-something' argument.");
                    usage();

                    // Else regard the argument as a xmlFileName.
                } else if (!(argument.matches(".*\\.xml") || argument.matches(
                        ".*\\" + ".xslt"))) {
                    System.out.println(
                            "File arguments must have extension .xml or .xslt");
                    usage();
                } else {
                    if (argument.matches(".*\\.xml")) {
                        xmlFileName = argument;

                    } else if (argument.matches(".*\\.xslt")) {
                        xsltFileName = argument;
                    } else {
                        throw new UnexpectedException("Unexpected argument: " + argument);
                    }
                }

            }
        }

        xmlPathSlashFileName = FileOperations.convertToPathSlashFileName(
                xmlFileName);
        System.out.format("xmlPathSlashFileName:%n    %s%nxsltPathSlashFileName:%n    " +
                                "%s%n",
                        xmlPathSlashFileName,
                        xsltPathSlashFileName);
        xsl.XslMain.setupAndCallTransformer(xmlPathSlashFileName, xsltFileName);
    }


    static private void usage() {
        System.out.println("Usage: JaxpDemo <file.xml> <file.xslt>");
        System.out.println("       -usage or -help: this message");
        System.out.println("       [But run from IntelliJ");
        System.exit(1);
    }
}
```

C:\Users\John\Documents\Sda\Code\BjaxXsl\Examples\XmlAviationWeather\JaxpDemo\src\JaxpDemo.java

Use SAX or DOM to parse an XML file and return the document. Pass the document along to do the XSL (XSLT) work.

```
public class XslMain {

    static final String OUTPUT_ENCODING = "UTF-8";
    static final boolean VALIDATE = true;

    static final String JAXP_SCHEMA_LANGUAGE = "http://java.sun" + "" +
            ".com/xml/jaxp/properties/schemaLanguage";
    static final String W3C_XML_SCHEMA = "http://www.w3.org/2001/XMLSchema";

    static final String JAXP_SCHEMA_SOURCE = "http://java.sun" +
            ".com/xml/jaxp/properties/schemaSource";

    public static void setupAndCallTransformer(
            String xmlPathSlashFileName, String xsltFileName) throws
            ParserConfigurationException,
            IOException, SAXException {

        String schemaSource =
                "takeOffAndLandingWeatherReports.xsd";
        DocumentBuilderFactory dbf = DocumentBuilderFactory.newInstance();

        dbf.setNamespaceAware(true);

        // Validation
        dbf.setValidating(VALIDATE);
        if (VALIDATE) {
            try {
                dbf.setAttribute(JAXP_SCHEMA_LANGUAGE, W3C_XML_SCHEMA);
```

```
            } catch (IllegalArgumentException e) {
                System.err.println(
                        "Error: JAXP DocumentBuilderFactory attribute " + "not " +
                                "recognized: " + JAXP_SCHEMA_LANGUAGE);
                System.err.println("Check to see if parser conforms to JAXP spec.");
                System.exit(1);
            }

            if (schemaSource != null) {
                dbf.setAttribute(JAXP_SCHEMA_SOURCE, new File(schemaSource));
            }
        }

        // Lexical Controls
        dbf.setIgnoringComments(false);
        dbf.setIgnoringElementContentWhitespace(true); // Only operates when validating.
        dbf.setCoalescing(false);
        dbf.setExpandEntityReferences(false);

        DocumentBuilder documentBuilder = dbf.newDocumentBuilder();

        // Set Error Handler
        OutputStreamWriter errorWriter = new OutputStreamWriter(System.err,
                                                                OUTPUT_ENCODING);
        documentBuilder.setErrorHandler(
                new dom.CentralDomErrorHandler(new PrintWriter(errorWriter, true)));
        Document document = documentBuilder.parse(xmlPathSlashFileName);

        File xsltFile = new File(xsltFileName);

        // Do work with the DOM document
        MyXslTransformer myXslTransformer = new MyXslTransformer(document, xsltFile);
    }
}
```

See (Oracle, 2017) http://docs.oracle.com/javase/tutorial/jaxp/xslt/writingDom.html,
See SAX Coding procedure or DOM Coding Procedure in this document.
C:\Users\John\Documents\Sda\Code\BjaxXsl\Examples\XmlAviationWeather\JaxpDemo\src\xsl\XslMain.java

Do the hookey pokey with TransformerFactory and Transformer to preform an identity transformation on your XML document, as an intermediate coding step to ensure everything is going your way.

```
public class MyXslTransformer {

    MyXslTransformer(Document document, File xsltFile) {
        TransformerFactory transformerFactory = TransformerFactory.newInstance();

        try {

            Transformer transformer = transformerFactory.newTransformer();

            // http://www.w3.org/TR/xslt#output
            transformer.setOutputProperty(OutputKeys.INDENT, "yes");
            transformer.setOutputProperty(OutputKeys.OMIT_XML_DECLARATION, "yes");

            DOMSource domSource = new DOMSource(document);
            StreamResult streamResult = new StreamResult(System.out);

            transformer.transform(domSource, streamResult);

        } catch (TransformerConfigurationException e) {
            System.out.println("* Transformer Factory error " + e.getMessage());

            Throwable throwable = e;
            if (e.getException() != null) {
                throwable = e.getException();
            }
            throwable.printStackTrace();
        } catch (TransformerException e) {
            System.out.println("* Transformation  error " + e.getMessage());

            Throwable throwable = e;
```

```
            if (e.getException() != null) {
                throwable = e.getException();
            }
            throwable.printStackTrace();
        }
    }
}
```

See (Oracle, 2017) http://docs.oracle.com/javase/tutorial/jaxp/xslt/writingDom.html, http://www.w3.org/TR/xslt#output
C:\Users\John\Documents\Sda\Code\BjaxXsl\Examples\XmlAviationWeather\JaxpDemo\src\xsl\MyXslTransformer.java

Refactor code to transform with an XSLT file.

```
public class MyXslTransformer {

    MyXslTransformer(Document document, File xsltFile) {
        TransformerFactory transformerFactory = TransformerFactory.newInstance();

        try {
            StreamSource xsltStreamSource = new StreamSource(xsltFile);
            Transformer transformer = transformerFactory.newTransformer(xsltStreamSource);

            // http://www.w3.org/TR/xslt#output
            transformer.setOutputProperty(OutputKeys.INDENT, "yes");
            transformer.setOutputProperty(OutputKeys.OMIT_XML_DECLARATION, "yes");

            NodeList list = document.getElementsByTagName("report");
            Node node = list.item(1); // The second report node.

            //            DOMSource domSource = new DOMSource(document);
            DOMSource domSource = new DOMSource(node);
            StreamResult streamResult = new StreamResult(System.out);

            transformer.transform(domSource, streamResult);

        } catch (TransformerConfigurationException e) {
            System.out.println("* Transformer Factory error " + e.getMessage());

            Throwable throwable = e;
            if (e.getException() != null) {
                throwable = e.getException();
            }
            throwable.printStackTrace();
        } catch (TransformerException e) {
            System.out.println("* Transformation  error " + e.getMessage());

            Throwable throwable = e;
            if (e.getException() != null) {
                throwable = e.getException();
            }
            throwable.printStackTrace();
        }
    }
}
```

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/jaxp/xslt/transformingXML.html
C:\Users\John\Documents\Sda\Code\BjaxXsl\Examples\XmlAviationWeather\JaxpDemo\src\xsl\MyXslTransformer.java

XSLT Coding Options

Transform a subtree of the DOM document, using getElementByTagName.

```
MyXslTransformer(Document document) {
    TransformerFactory transformerFactory = TransformerFactory.newInstance();

        try {
```

```
            Transformer transformer = transformerFactory.newTransformer();

            // http://www.w3.org/TR/xslt#output
            transformer.setOutputProperty(OutputKeys.INDENT, "yes");
            transformer.setOutputProperty(OutputKeys.OMIT_XML_DECLARATION, "yes");

            NodeList list = document.getElementsByTagName("report");
            Node node = list.item(1); // The second report node.

            //            DOMSource domSource = new DOMSource(document);
            DOMSource domSource = new DOMSource(node);
            StreamResult streamResult = new StreamResult(System.out);

            transformer.transform(domSource, streamResult);
...
// Output (neither reports for YSSY nor YSCB shown).
<report xmlns="http://tempuri.org/TakeOffAndLandingWeatherReports.xsd">
<location>YMML</location>
<dateTimeUtc>2003-05-16T16:00:00Z</dateTimeUtc>
<wind>
<direction reference="magnetic">060</direction>
<speed unit="knots">14</speed>
</wind>
<qnh unit="hectoPascals">1013.2</qnh>
<temperature unit="celsius">05</temperature>
<dewPoint unit="celsius">02</dewPoint>
<sky>
<cavok/>
</sky>
<remarks>Runway 16L closed for repair.</remarks>
</report>
```

See (Oracle, 2012) http://docs.oracle.com/javase/tutorial/jaxp/xslt/writingDom.html
C:\Users\John\Documents\Sda\Code\BjaxXsl\Examples\XmlAviationWeather\JaxpDemo\src\xsl\MyXslTransformer.java

Output result to a file rather than System.out

```
public class MyXslTransformer {

    MyXslTransformer(Document document, File xsltFile) {
        TransformerFactory transformerFactory = TransformerFactory.newInstance();

        try {
            StreamSource xsltStreamSource = new StreamSource(xsltFile);
            Transformer transformer = transformerFactory.newTransformer(xsltStreamSource);

            // http://www.w3.org/TR/xslt#output
            transformer.setOutputProperty(OutputKeys.INDENT, "yes");
            transformer.setOutputProperty(OutputKeys.OMIT_XML_DECLARATION, "yes");

            NodeList list = document.getElementsByTagName("report");
            Node node = list.item(1); // The second report node.

            //            DOMSource domSource = new DOMSource(document);
            DOMSource domSource = new DOMSource(node);
//            StreamResult streamResult = new StreamResult(System.out);

            PrintWriter streamOut = new PrintWriter(new FileWriter("MyXsltOutput.html"));
            StreamResult streamResult = new StreamResult(streamOut);

            transformer.transform(domSource, streamResult);

        } catch (TransformerConfigurationException e) {
            System.out.println("* Transformer Factory error " + e.getMessage());

            Throwable throwable = e;
            if (e.getException() != null) {
                throwable = e.getException();
            }
            throwable.printStackTrace();
        } catch (TransformerException e) {
            System.out.println("* Transformation  error " + e.getMessage());
```

```
            Throwable throwable = e;
            if (e.getException() != null) {
                throwable = e.getException();
            }
            throwable.printStackTrace();
        } catch (IOException e) {
            e.printStackTrace();
        }
    }
}
```

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/jaxp/xslt/transformingXML.html
C:\Users\John\Documents\Sda\Code\BjaxXsl\Examples\XmlAviationWeather\JaxpDemo\src\xsl\MyXslTransformer.java

Arbitrary Data to XML with javax.xml.transform and SAX

Take arbitrary data (in a text file, database, or network stream) and write it out as XML using SAX.

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/jaxp/xslt/generatingXML.html

Coding procedure:

Create a data source in some arbitrary/proprietary format.

```
location~ YMML
dateTimeUtc~ 2003-05-16T16:00:00Z
wind~ 06014
direction~ 060
speed~ 14
qnh~ 1013.2
temperature~ 05
dewPoint~ 02
sky~
cavok~
remarks~ Runway 16L closed for repair.
```

C:\Users\John\Documents\Sda\Code\BjaxXsl\Examples\XmlAviationWeather\XmlSource\takeOffAndLandingWeatherReportValuePairs.txt

Implement org.xml.sax.XMLReader. Implement the parse method that takes a org.xml.sax.InputSource. Raise events to a content handler, passing relevant data.

```
public class WeatherReportTextFileReader implements XMLReader {

    ContentHandler mContentHandler = null;

    String mNamespaceUri = "";
    String mRootElement = "report";
    AttributesImpl mAttributes = new AttributesImpl();

    String mIndent = "\n  ";

    @Override
    public void parse(InputSource inputSource) throws IOException, SAXException {
        BufferedReader bufferedReader = new BufferedReader(
                inputSource.getCharacterStream());
        Scanner scannerInputSource = new Scanner(bufferedReader);
        try {

            if (mContentHandler == null) {
                throw new SAXException("No content handler");
            }

            // Insert newline
```

```
            mContentHandler.ignorableWhitespace("\n".toCharArray(), 0, 1);
            mContentHandler.startDocument();
            mContentHandler.startElement(mNamespaceUri, mRootElement, mRootElement,
                                         mAttributes);
            // Any number of spaces surrounding a tilde "~" or any number of spaces
            // till End cool of Line.
            String field = "";
            String value = "";

            while (scannerInputSource.hasNextLine()) {
                String line = scannerInputSource.nextLine();
                String[] lineTokens = line.split("~");
                if (lineTokens.length == 1) {
                    field = lineTokens[0];
                    value = "";
                } else if (lineTokens.length == 2) {
                    field = lineTokens[0];
                    value = lineTokens[1].trim();
                } else {
                    throw new UnexpectedException(
                            "lineTokens has unexpected length: " + lineTokens.length);
                }

                if (field.matches("qnh")) {
                    mAttributes.addAttribute(mNamespaceUri, "unit", "unit", "string",
                                             "hectoPascals");
                }
                xmlElementOutput(field, value, mAttributes);
                while (mAttributes.getLength() != 0) {
                    mAttributes.removeAttribute(0);
                }
            }

            mContentHandler.ignorableWhitespace("\n".toCharArray(), 0, 1);
            mContentHandler.endElement(mNamespaceUri, mRootElement, mRootElement);
            mContentHandler.endDocument();

        } catch (UnexpectedException e) {
            e.printStackTrace();
        }
    }

    private void xmlElementOutput(String elementLocalName, String elementTextNodeValue,
                                  AttributesImpl attributes) throws SAXException {
        mContentHandler.ignorableWhitespace(mIndent.toCharArray(), 0, mIndent.length());

        mContentHandler.startElement(mNamespaceUri, elementLocalName, elementLocalName
        /* qName */, attributes);
        mContentHandler.characters(elementTextNodeValue.toCharArray(), 0,
                                   elementTextNodeValue.length());
        mContentHandler.endElement(mNamespaceUri, elementLocalName, elementLocalName);
    }
```

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/jaxp/xslt/generatingXML.html
C:\Users\John\Documents\Sda\Code\BjaxXsl\Examples\XmlAviationWeather\JaxpDemo\src\sax\text\reader\WeatherReportTextFileReader.java

Enable the content handler to be registered.

```
    @Override
    public void setContentHandler(ContentHandler contentHandler) {
        this.mContentHandler = contentHandler;
    }

    @Override
    public ContentHandler getContentHandler() {
        return this.mContentHandler;
    }
```

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/jaxp/xslt/generatingXML.html
C:\Users\John\Documents\Sda\Code\BjaxXsl\Examples\XmlAviationWeather\JaxpDemo\src\sax\text\reader\WeatherReportTextFileReader.java

Implement the rest of the XMLReader methods using default stubs if you aren't interested in any specialized handling.

```
    @Override
    public void parse(String s) throws IOException, SAXException {
    }

    @Override
    public void setErrorHandler(ErrorHandler errorHandler) {
    }

    @Override
    public ErrorHandler getErrorHandler() {
        return null;
    }
....
```

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/jaxp/xslt/generatingXML.html
C:\Users\John\Documents\Sda\Code\BjaxXsl\Examples\XmlAviationWeather\JaxpDemo\src\sax\text\reader\WeatherReportTextFileReader.java

Create client code to call the above custom XMLReader. You need to reference: the Data input source; the custom XML reader; an output source (e.g. filename or screen output); mix these together with a transformer.

```
private enum OutputDirection {
    FILE,
    SCREEN
}

static private void WriteXmlFromArbitraryDataStructure(String txtFileName,
                                                       OutputDirection
                                                               outputDirection)
        throws UnexpectedException {
    WeatherReportTextFileReader saxReader = new WeatherReportTextFileReader();
    File txtFile = new File(txtFileName);

    BufferedReader bufferedReader = null;
    try {
        bufferedReader = new BufferedReader(new FileReader(txtFile));

    } catch (FileNotFoundException e) {
        e.printStackTrace();
    }
    InputSource inputSource = new InputSource(bufferedReader);
    SAXSource saxSource = new SAXSource(saxReader, inputSource);

    StreamResult streamResult = null;
    switch (outputDirection) {
        case FILE:
            try {
                FileOutputStream fileOutputStream = new FileOutputStream(
                        "takeOffAndLandingWeatherReportResult.xml");
                streamResult = new StreamResult(fileOutputStream);
            } catch (FileNotFoundException e) {
                e.printStackTrace();
            }
            break;

        case SCREEN:
            streamResult = new StreamResult(System.out);
            break;

        default:
            throw new UnexpectedException(
                    "Unexpected outputDirection: " + outputDirection.toString());
    }

    TransformerFactory transformerFactory = TransformerFactory.newInstance();
    Transformer transformer = null;
    try {
        transformer = transformerFactory.newTransformer();
```

```
        transformer.setOutputProperty(OutputKeys.OMIT_XML_DECLARATION, "no");
        transformer.transform(saxSource, streamResult);
    } catch (TransformerConfigurationException e) {
        e.printStackTrace();
    } catch (TransformerException e) {
        e.printStackTrace();
    }
}
```

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/jaxp/xslt/generatingXML.html
C:\Users\John\Documents\Sda\Code\BjaxXsl\Examples\XmlAviationWeather\JaxpDemo\src\sax\text\reader\WeatherReportTextFileReader.java

Streaming API for XML (StAX)

The Stax API entails choosing between two API sets: the Cursor API and an Iterator API. The differences:

	Cursor API	**Iterator API**
Descripton	High performance forward only walk through XML document.	Pull XML events from XML source.
Interfaces	XMLStreamReader XMLStreamWriter	XMLEventReader XMLEventWriter
Choosing	• Best if memory constrained. • Best if high performance is the highest priority.	• Recommended, all else being equal. • Has more features for processing XML. (??)

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/jaxp/stax/api.html

Only the Cursor API can write empty elements as a single tag (a "self closing tag").

```
// Outputting an empty element with a single tag, <example/>,
// is possible with the Cursor API usingXMLStreamWriter:

xmlStreamWriter.writeEmptyElement("example");

// Outputting an empty element with a single tag, <example/>,
// is not possible with the Iterator API using XMLEventWriter,
// as far as I know. In this case you're stuck with producing an empty
// element with two tags <example></example>:

xmlEventWriter.add(xmlEventFactory.createStartElement("", null, "example"));
xmlEventWriter.add(xmlEventFactory.createEndElement("", null, "example"));
```

http://stackoverflow.com/a/29775017/872154 "XMLEventWriter: how can I tell it to write empty elements?", Answer by John Bentley

Stax Cursor API

Overview

The Stax cursor API has two main interfaces, XMLStreamReader and XMLStreamWriter, for you to use.

```
public interface XMLStreamReader {
    public int next() throws XMLStreamException;
    public boolean hasNext() throws XMLStreamException;

    public String getText();
    public String getLocalName();
    public String getNamespaceURI();
    // ... other methods not shown
```

```
}

public interface XMLStreamWriter {
    public void writeStartElement(String localName) throws XMLStreamException;
    public void writeEndElement() throws XMLStreamException;
    public void writeCharacters(String text) throws XMLStreamException;
    // ... other methods not shown
}
```

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/jaxp/stax/api.html

Stax Cursor API - Read XML

Setup and call Stax Cursor Reader.

```
static public void setupAndCallStaxCursorXmlReader(String xmlFileName,
                                                   String xsdFileName) throws
        UnexpectedException {

    XMLInputFactory xmlInputFactory = XMLInputFactory.newInstance();
    XMLStreamReader xmlStreamReader = null;
    FileInputStream inputStream = null;

    xmlInputFactory.setProperty(XMLInputFactory.IS_COALESCING, true);
    xmlInputFactory.setProperty(XMLInputFactory.IS_REPLACING_ENTITY_REFERENCES, true);
    xmlInputFactory.setProperty(XMLInputFactory.IS_SUPPORTING_EXTERNAL_ENTITIES,
                                true);
...

    try {
        inputStream = new FileInputStream(xmlFileName);
        xmlStreamReader = xmlInputFactory.createXMLStreamReader(inputStream, "UTF-8");
    } catch (XMLStreamException e) {
        e.printStackTrace();
    } catch (FileNotFoundException e) {
        e.printStackTrace();
    }

    outputStaxCursorXmlReader(xmlStreamReader);
}
```

C:\Users\John\Documents\Sda\Code\BjaxXsl\Examples\XmlAviationWeather\JaxpDemo\src\stax\StaxMain.java

Stax Cursor API: Using the XMLStreamReader to parse the XML source and write some (arbitrarily formatted) text output.

```
static private void printIndent(PrintStream outStream, int indentLevel) {
    String BASIC_INDENT = "  ";
    for (int i = 0; i < indentLevel; i++) {
        outStream.print(BASIC_INDENT);
    }
}

static private void outputStaxCursorXmlReader(XMLStreamReader xmlStreamReader) {
    int indentLevel = 0;
    PrintStream outStream = System.out;

    try {
        while (xmlStreamReader.hasNext()) {
            int eventType = xmlStreamReader.next();
            switch (eventType) {
                case XMLStreamConstants.START_ELEMENT:
                    outStream.println();
                    if (xmlStreamReader.getLocalName().equals("report")) {
                        outStream.println();
                    }
                    printIndent(outStream, indentLevel++);
                    outStream.print(xmlStreamReader.getLocalName());

                    for (int i = 0; i < xmlStreamReader.getAttributeCount(); i++) {
```

```
                        outStream.printf(" [%s=%s]",
                                        xmlStreamReader.getAttributeLocalName(i),
                                        xmlStreamReader.getAttributeValue(i));
                    }
                    outStream.print(": ");
                    break;

                case XMLStreamConstants.END_ELEMENT:
                    indentLevel--;
                    break;
                case XMLStreamConstants.CHARACTERS:
                    if (!xmlStreamReader.isWhiteSpace()) {
                        outStream.print(xmlStreamReader.getText());
                    }

                    break;
                default:
                    // Ignore other eventTypes.
            }
        }
    } catch (XMLStreamException e) {
        e.printStackTrace();
    }
}
```

C:\Users\John\Documents\Sda\Code\BjaxXsl\Examples\XmlAviationWeather\JaxpDemo\src\stax\StaxXmlReader.java
(Oracle, 2017) http://docs.oracle.com/javase/tutorial/jaxp/stax/using.html

The input and output ...

```
// // Example input
<?xml version="1.0" ?>
<!--<?xml-stylesheet type="text/xsl" href="NameValuePairs.xslt" ?>-->
<takeOffAndLandingWeatherReports
xmlns="http://tempuri.org/TakeOffAndLandingWeatherReports.xsd"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://tempuri.org/TakeOffAndLandingWeatherReports.xsd
TakeOffAndLandingWeatherReports.xsd">

  <!-- A series of weather reports for various locations and date/times -->
  <report>
    <location>YSSY</location>
    <dateTimeUtc>2003-10-23T15:30:00Z</dateTimeUtc>

    <wind>
      <direction reference="magnetic">060</direction>
      <speed unit="knots">23</speed>
    </wind>

    <qnh unit="hectoPascals">1013.12</qnh>
    <temperature unit="celsius">25</temperature>
    <dewPoint unit="celsius">-02</dewPoint>

    <sky>
      <cloud cover="few (0-2 eights)">
        <height unit="feet" reference="agl">4000</height>
      </cloud>

      <cloud cover="broken (5-7 eights)">
        <height unit="feet" reference="agl">11000</height>
      </cloud>
      <ceiling cover="broken (5-7 eights)">
        <height unit="feet" reference="agl">4000</height>
      </ceiling>

      <visibility>Greater than 10 kilometres.</visibility>

      <phenomenon intensity="light" action="showers">rain</phenomenon>
      <phenomenon intensity="medium" action="thunderstorm">hail</phenomenon>
    </sky>
  </report>

  <report>
```

```
    <location>YMML</location>
...

// Example output
xmlFileName:
    takeOffAndLandingWeatherReports.xml
xsdFileName:
    takeOffAndLandingWeatherReports.xsd

takeOffAndLandingWeatherReports
[schemaLocation=http://tempuri.org/TakeOffAndLandingWeatherReports.xsd
TakeOffAndLandingWeatherReports.xsd]:

  report:
    location: YSSY
    dateTimeUtc: 2003-10-23T15:30:00Z
    wind:
      direction [reference=magnetic]: 060
      speed [unit=knots]: 23
    qnh [unit=hectoPascals]: 1013.12
    temperature [unit=celsius]: 25
    dewPoint [unit=celsius]: -02
    sky:
      cloud [cover=few (0-2 eights)]:
        height [unit=feet] [reference=agl]: 4000
      cloud [cover=broken (5-7 eights)]:
        height [unit=feet] [reference=agl]: 11000
      ceiling [cover=broken (5-7 eights)]:
        height [unit=feet] [reference=agl]: 4000
      visibility: Greater than 10 kilometres.
      phenomenon [intensity=light] [action=showers]: rain
      phenomenon [intensity=medium] [action=thunderstorm]: hail

  report:
    location: YMML
    dateTimeUtc: 2003-05-16T16:00:00Z
    wind:
      direction: 060
...
```

C:\Users\John\Documents\Sda\Code\BjaxXsl\Examples\XmlAviationWeather\JaxpDemo\src\stax\StaxXmlReader.java

Validate a XML document before using the Stax Cursor API to parse the XML document.

```
public class StaxXmlReader {
    private enum XmlErrorCheckingLevel {WELL_FORMED, VALID}

    private static final XmlErrorCheckingLevel mActiveXmlErrorCheckingLevel =
            XmlErrorCheckingLevel.WELL_FORMED;

    static public void setupAndCallStaxCursorXmlReader(String xmlFileName,
                                                       String xsdFileName) throws
            UnexpectedException {

        XMLInputFactory xmlInputFactory = XMLInputFactory.newInstance();
        XMLStreamReader xmlStreamReader = null;
        FileInputStream inputStream = null;

        xmlInputFactory.setProperty(XMLInputFactory.IS_COALESCING, true);
        xmlInputFactory.setProperty(XMLInputFactory.IS_REPLACING_ENTITY_REFERENCES, true);
        xmlInputFactory.setProperty(XMLInputFactory.IS_SUPPORTING_EXTERNAL_ENTITIES,
                                    true);

        // A module level switch.
        switch (mActiveXmlErrorCheckingLevel) {
            case WELL_FORMED:
                // Do nothing. XML documents that are not well formed will throw an
                // XMLStreamException as they are parsed.
                break;
            case VALID:
                // Not the following produces an error 'true value of isValidating not
                // supported'
                // So we don't use it.
```

```
                // xmlInputFactory.setProperty(XMLInputFactory.IS_VALIDATING, true);

                StaxValidate(xmlFileName, xsdFileName);
                break;
            default:
                throw new UnexpectedException("Unexpected Switch case reached");
        }

        try {
            inputStream = new FileInputStream(xmlFileName);
            xmlStreamReader = xmlInputFactory.createXMLStreamReader(inputStream, "UTF-8");
        } catch (XMLStreamException e) {
            e.printStackTrace();
        } catch (FileNotFoundException e) {
            e.printStackTrace();
        }

        outputStaxCursorXmlReader(xmlStreamReader);
    }

    static private void StaxValidate(String xmlFileName, String xsdFileName) {
        SchemaFactory schemaFactory = SchemaFactory.newInstance(XMLConstants

.W3C_XML_SCHEMA_NS_URI);
        try {
            Schema schema = schemaFactory.newSchema(new File(xsdFileName));
            Validator validator = schema.newValidator();

            StreamSource streamSource = new StreamSource(xmlFileName);
            validator.validate(streamSource);
        } catch (SAXParseException e) {
            System.err.printf("%s is invalid (against %s) (line %d, " +
                                    "column" +
                                    " %d):%n %s",
                              xmlFileName,
                              xsdFileName,
                              e.getLineNumber(),
                              e.getColumnNumber(),
                              e.getMessage());
            System.exit(-1); // Terminate with error.
        } catch (SAXException e) {
            e.printStackTrace();
            System.exit(-1); // Terminate with error.
        } catch (IOException e) {
            e.printStackTrace();
            System.exit(-1); // Terminate with error.
        }
    }
```

C:\Users\John\Documents\Sda\Code\BjaxXsl\Examples\XmlAviationWeather\JaxpDemo\src\stax\StaxMain.java

Stax Cursor API - Write XML

Take a text file of arbitrary format and output it as XML using Stax (Cursor API/XMLStreamWriter):

Setup a scanner for the input file.

```
static public void CursorApiWriter(String txtFileName,
                                   OutputDirection outputDirection) throws
        UnexpectedException {

    // Setup a scanner for the input file
    File inputFile = new File(txtFileName);
    BufferedReader bufferedReader = null;
    Scanner scanner = null;
    try {
        bufferedReader = new BufferedReader(new FileReader(inputFile));
        scanner = new Scanner(bufferedReader);

    } catch (FileNotFoundException e) {
        e.printStackTrace();
```

```
    }
```

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/jaxp/stax/using.html
C:\Users\John\Documents\Sda\Code\BjaxXsl\Examples\XmlAviationWeather\JaxpDemo\src\stax\StaxXmlWriter.java

Create a XMLStreamWriter for your desired output stream (e.g. file or screen).

```
    // Create a XMLStreamWriter for your desired output stream (e.g. file or screen).
    XMLOutputFactory xmlOutputFactory = XMLOutputFactory.newInstance();
    XMLStreamWriter xmlStreamWriter = null;
    try {
        switch (outputDirection) {
            case FILE:
                final String outputFileName = "takeOffAndLandingWeatherReportResult" +
                        ".xml";
                try {
                    FileOutputStream fileOutputStream = new FileOutputStream(
                            outputFileName );
                    xmlStreamWriter = xmlOutputFactory.createXMLStreamWriter(
                            fileOutputStream);
                    System.out.println("outputFileName: " + outputFileName);
                } catch (FileNotFoundException e) {
                    e.printStackTrace();
                }
                break;

            case SCREEN:
                xmlStreamWriter = xmlOutputFactory.createXMLStreamWriter(System.out);
                break;

            default:
                throw new UnexpectedException("Unexpected outputDirection: " +
                                                    outputDirection.toString());
        }
    } catch (XMLStreamException e) {
        e.printStackTrace();
    }
```

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/jaxp/stax/using.html
C:\Users\John\Documents\Sda\Code\BjaxXsl\Examples\XmlAviationWeather\JaxpDemo\src\stax\StaxXmlWriter.java

Process the input file using a scanner. In this example we use line by line operations. As you process the input file write to XML using the XML writer.

```
    try {
        String field;
        String value;
        // Find one or more characters that are not "~" and not a space.
        String regexPattern = "[^~\\s]+";
        xmlStreamWriter.writeStartDocument();
        xmlStreamWriter.writeCharacters("\r\n");
        xmlStreamWriter.writeStartElement("report"); // Root element.
        xmlStreamWriter.writeCharacters("\r\n");
        while (scanner.hasNextLine()) {
            field = scanner.findInLine(regexPattern);
            // findInLine consumes that which it matches. So we can apply the same
            // regex pattern to consume more of the line.
            value = scanner.findInLine(regexPattern);
            scanner.nextLine();
            xmlStreamWriter.writeCharacters("  ");
            if (field.equals("dateTimeUtc")) {
                // Insert value as an attribute
                xmlStreamWriter.writeEmptyElement(field);
                xmlStreamWriter.writeAttribute("Zulu", value);
            } else {
                // Insert value as an element character
                xmlStreamWriter.writeStartElement(field);
                xmlStreamWriter.writeCharacters(value);
```

```
                xmlStreamWriter.writeEndElement();
            }
            xmlStreamWriter.writeCharacters("\r\n");
        } // while
        xmlStreamWriter.writeEndElement(); // "Report" root element.
        xmlStreamWriter.writeEndDocument();
        xmlStreamWriter.flush();
        if (xmlStreamWriter != null) {
                xmlStreamWriter.close();
            System.out.println("Output success.");
        } // Try
    } catch (XMLStreamException e) {
        e.printStackTrace();
    } finally {
        if (scanner != null) {
            scanner.close();
        }
    }
}
```

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/jaxp/stax/using.html
C:\Users\John\Documents\Sda\Code\BjaxXsl\Examples\XmlAviationWeather\JaxpDemo\src\stax\StaxXmlWriter.java

Example input and output.

```
// Input.  takeOffAndLandingWeatherReportValuePairs.txt
location~ YMMLX
dateTimeUtc~ 2003-05-16T16:00:00Z
wind~ 06014
direction~ 060
speed~ 14
qnh~ 1013.2
temperature~ 05
dewPoint~ 02
sky~
cavok~
remarks~ Runway 16L closed for repair.

// Output. takeOffAndLandingWeatherReportResult.xml (when choosing the file output option)
<?xml version="1.0" ?>
<report>
  <location>YMMLX</location>
  <dateTimeUtc Zulu="2003-05-16T16:00:00Z"/>
  <wind>06014</wind>
  <direction>060</direction>
  <speed>14</speed>
  <qnh>1013.2</qnh>
  <temperature>05</temperature>
  <dewPoint>02</dewPoint>
  <sky></sky>
  <cavok></cavok>
  <remarks>Runway</remarks>
</report>
```

Stax Iterator API

Stax Iterator API - Read XML

Setup and all Stax Iterator Reader (use XMLEventReader).

```
static public void setupAndCallStaxIteratorXmlReader(String xmlFileName,
                                                      String xsdFileName) throws
        UnexpectedException {

    XMLInputFactory xmlInputFactory = XMLInputFactory.newInstance();
    XMLEventReader xmlEventReader = null;
    FileInputStream inputStream = null;

    xmlInputFactory.setProperty(XMLInputFactory.IS_COALESCING, true);
```

```
    xmlInputFactory.setProperty(XMLInputFactory.IS_REPLACING_ENTITY_REFERENCES, true);
    xmlInputFactory.setProperty(XMLInputFactory.IS_SUPPORTING_EXTERNAL_ENTITIES,
                                true);

...

    try {
        inputStream = new FileInputStream(xmlFileName);
        xmlEventReader = xmlInputFactory.createXMLEventReader(inputStream, "UTF-8");
    } catch (XMLStreamException e) {
        e.printStackTrace();
    } catch (FileNotFoundException e) {
        e.printStackTrace();
    }

    outputStaxIteratorXmlReader(xmlEventReader);
}
```

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/jaxp/stax/using.html
C:\Users\John\Documents\Sda\Code\BjaxXsl\Examples\XmlAviationWeather\JaxpDemo\src\stax\StaxXmlIteratorReader.java

Stax Iterator API: Use the XMLEventReader to parse XML source and write some arbitrarily formatted text output.

```
static private void printIndent(PrintStream outStream, int indentLevel) {
    String BASIC_INDENT = "  ";
    for (int i = 0; i < indentLevel; i++) {
        outStream.print(BASIC_INDENT);
    }
}

static private void outputStaxIteratorXmlReader(XMLEventReader xmlEventReader) {
    int indentLevel = 0;
    PrintStream outStream = System.out;
    XMLEvent xmlEvent = null;

    try {
        while (xmlEventReader.hasNext()) {
            xmlEvent = xmlEventReader.nextEvent();

            switch (xmlEvent.getEventType()) {
                case XMLStreamConstants.START_ELEMENT:
                    outStream.println();
                    StartElement startElement = (StartElement) xmlEvent;
                    if (startElement.getName().getLocalPart().equals("report")) {
                        outStream.println();
                    }
                    printIndent(outStream, indentLevel++);
                    outStream.print(startElement.getName().getLocalPart());

                    Iterator iterator = startElement.getAttributes();
                    while (iterator.hasNext()) {
                        Attribute attribute = (Attribute) iterator.next();
                        outStream.printf(" [%s=%s]",
                                         attribute.getName(),
                                         attribute.getValue());
                    }

                    outStream.print(": ");
                    break;

                case XMLStreamConstants.END_ELEMENT:
                    indentLevel--;
                    break;
                case XMLStreamConstants.CHARACTERS:
                    Characters characters = (Characters) xmlEvent;
                    if (!characters.isWhiteSpace()) {
                        outStream.print(characters.getData());
                    }

                    break;
                default:
                    // Ignore other eventTypes.
```

```
            } // Switch
        } // While
    } catch (XMLStreamException e) {
        e.printStackTrace();
    }
}
```

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/jaxp/stax/using.html
C:\Users\John\Documents\Sda\Code\BjaxXsl\Examples\XmlAviationWeather\JaxpDemo\src\stax\StaxXmlIteratorReader.java

Input and output.

```
// takeOffAndLandingWeatherReports.xml
<?xml version="1.0" ?>
<!--<?xml-stylesheet type="text/xsl" href="NameValuePairs.xslt" ?>-->
<takeOffAndLandingWeatherReports
xmlns="http://tempuri.org/TakeOffAndLandingWeatherReports.xsd"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://tempuri.org/TakeOffAndLandingWeatherReports.xsd
TakeOffAndLandingWeatherReports.xsd">

  <!--<takeOffAndLandingWeatherReports
      xmlns="http://tempuri.org/TakeOffAndLandingWeatherReports.xsd" >-->

  <!-- A series of weather reports for various locations and date/times -->
  <report>
    <location>YSSY</location>
    <dateTimeUtc>2003-10-23T15:30:00Z</dateTimeUtc>
    <fonzi>sdfkj</fonzi>
    <wind>
      <direction reference="magnetic">060</direction>
      <speed unit="knots">23</speed>
    </wind>

    <qnh unit="hectoPascals">1013.12</qnh>
    <temperature unit="celsius">25</temperature>
    <dewPoint unit="celsius">-02</dewPoint>

    <sky>
      <cloud cover="few (0-2 eights)">
        <height unit="feet" reference="agl">4000</height>
      </cloud>

      <cloud cover="broken (5-7 eights)">
        <height unit="feet" reference="agl">11000</height>
      </cloud>
      <ceiling cover="broken (5-7 eights)">
        <height unit="feet" reference="agl">4000</height>
      </ceiling>

      <visibility>Greater than 10 kilometres.</visibility>

      <phenomenon intensity="light" action="showers">rain</phenomenon>
      <phenomenon intensity="medium" action="thunderstorm">hail</phenomenon>
    </sky>
  </report>

  <report>
    <location>YMML</location>
      <cloud cover="few (0-2 eights)">
...

// Output
xmlFileName:
    takeOffAndLandingWeatherReports.xml
xsdFileName:
    takeOffAndLandingWeatherReports.xsd

takeOffAndLandingWeatherReports [{http://www.w3.org/2001/XMLSchema-
instance}schemaLocation=http://tempuri.org/TakeOffAndLandingWeatherReports.xsd
TakeOffAndLandingWeatherReports.xsd]:

  report:
```

```
    location: YSSY
    dateTimeUtc: 2003-10-23T15:30:00Z
    wind:
      direction [reference=magnetic]: 060
      speed [unit=knots]: 23
    qnh [unit=hectoPascals]: 1013.12
    temperature [unit=celsius]: 25
    dewPoint [unit=celsius]: -02
    sky:
      cloud [cover=few (0-2 eights)]:
        height [unit=feet] [reference=agl]: 4000
      cloud [cover=broken (5-7 eights)]:
        height [unit=feet] [reference=agl]: 11000
      ceiling [cover=broken (5-7 eights)]:
        height [unit=feet] [reference=agl]: 4000
      visibility: Greater than 10 kilometres.
      phenomenon [action=showers] [intensity=light]: rain
      phenomenon [action=thunderstorm] [intensity=medium]: hail

  report:
    location: YMML
    dateTimeUtc: 2003-05-16T16:00:00Z
...
```

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/jaxp/stax/using.html
C:\Users\John\Documents\Sda\Code\BjaxXsl\Examples\XmlAviationWeather\JaxpDemo\src\stax\StaxXmlIteratorReader.java

Validate the input XML document (against an XSD document) using Stax.

```
private enum XmlErrorCheckingLevel {WELL_FORMED, VALID}

private static final XmlErrorCheckingLevel mActiveXmlErrorCheckingLevel =
        XmlErrorCheckingLevel.VALID;

// Setup and all Stax Iterator Reader
static public void setupAndCallStaxIteratorXmlReader(String xmlFileName,
                                                     String xsdFileName) throws
        UnexpectedException {

    XMLInputFactory xmlInputFactory = XMLInputFactory.newInstance();
    XMLEventReader xmlEventReader = null;
    FileInputStream inputStream = null;

    xmlInputFactory.setProperty(XMLInputFactory.IS_COALESCING, true);
    xmlInputFactory.setProperty(XMLInputFactory.IS_REPLACING_ENTITY_REFERENCES, true);
    xmlInputFactory.setProperty(XMLInputFactory.IS_SUPPORTING_EXTERNAL_ENTITIES,
                                true);

    // A module level switch.
    switch (mActiveXmlErrorCheckingLevel) {
        case WELL_FORMED:
            // Do nothing. XML documents that are not well formed will throw an
            // XMLStreamException as they are parsed.
            break;
        case VALID:
            // Not the following produces an error 'true value of isValidating not
            // supported'
            // So we don't use it.
            // xmlInputFactory.setProperty(XMLInputFactory.IS_VALIDATING, true);

            StaxValidate(xmlFileName, xsdFileName);
            break;
        default:
            throw new UnexpectedException("Unexpected Switch case reached");
    }

    try {
        inputStream = new FileInputStream(xmlFileName);
        xmlEventReader = xmlInputFactory.createXMLEventReader(inputStream, "UTF-8");
    } catch (XMLStreamException e) {
        e.printStackTrace();
    } catch (FileNotFoundException e) {
        e.printStackTrace();
```

```
    }

    outputStaxIteratorXmlReader(xmlEventReader);
}

static private void StaxValidate(String xmlFileName, String xsdFileName) {
    SchemaFactory schemaFactory = SchemaFactory.newInstance(XMLConstants
                                                            .W3C_XML_SCHEMA_NS_URI);
    try {
        Schema schema = schemaFactory.newSchema(new File(xsdFileName));
        Validator validator = schema.newValidator();

        StreamSource streamSource = new StreamSource(xmlFileName);
        validator.validate(streamSource);
    } catch (SAXParseException e) {
        System.err.printf("%s is invalid (against %s) (line %d, " +
                                "column" +
                                " %d):%n %s",
                          xmlFileName,
                          xsdFileName,
                          e.getLineNumber(),
                          e.getColumnNumber(),
                          e.getMessage());
        System.exit(-1); // Terminate with error.
    } catch (SAXException e) {
        e.printStackTrace();
        System.exit(-1); // Terminate with error.
    } catch (IOException e) {
        e.printStackTrace();
        System.exit(-1); // Terminate with error.
    }
}
```

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/jaxp/stax/using.html
C:\Users\John\Documents\Sda\Code\BjaxXsl\Examples\XmlAviationWeather\JaxpDemo\src\stax\StaxXmlIteratorReader.java

Example validation input and output

```
// takeOffAndLandingWeatherReports.xml
<?xml version="1.0" ?>
<takeOffAndLandingWeatherReports
xmlns="http://tempuri.org/TakeOffAndLandingWeatherReports.xsd"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://tempuri.org/TakeOffAndLandingWeatherReports.xsd
TakeOffAndLandingWeatherReports.xsd">

  <!-- A series of weather reports for various locations and date/times -->
  <report>
    <location>YSSY</location>
    <dateTimeUtc>2003-10-23T15:30:00Z</dateTimeUtc>
    <!-- Invalid entry -->
    <fonzi>sdfkj</fonzi>
    <wind>
      <direction reference="magnetic">060</direction>
      <speed unit="knots">23</speed>
    </wind>
...

// takeOffAndLandingWeatherReports.xsd (a large schema document)
<?xml version="1.0" encoding="utf-8" ?>
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema"
targetNamespace="http://tempuri.org/TakeOffAndLandingWeatherReports.xsd"
xmlns="http://tempuri.org/TakeOffAndLandingWeatherReports.xsd"
xmlns:tns="http://tempuri.org/TakeOffAndLandingWeatherReports.xsd"
xmlns:xhtml="http://www.w3.org/1999/xhtml" elementFormDefault="qualified"
attributeFormDefault="unqualified" version="0.5">
  <!-- Note the default namespace is just there so that .NET studio
      can find this schema during development -->
  <xsd:import namespace="http://www.w3.org/1999/xhtml" />
  <xsd:annotation>
    <xsd:documentation xml:space="preserve">
            A Take Off and Landing Aviation Report contains a summary of the
            current weather conditions within 5nm of an aerodrome.
```

```
            Aircraft use it for take off and landing.

            Commonly broadcast as an Automatic Terminal Information Service (ATIS).

            See Aeronautical Information Publication Australia Gen 3.5 -8 for
            specifications from which this xml schema was composed.

            John Bentley
            This schema come from no official source.
            It is written as a means of practicing the writing of XSD Schemas
        </xsd:documentation>
  </xsd:annotation>

  <xsd:element name="takeOffAndLandingWeatherReports">
    <xsd:complexType>
      <xsd:sequence>
        <xsd:element name="report" type="reportType" maxOccurs="unbounded" />
      </xsd:sequence>
    </xsd:complexType>
  </xsd:element>
...

// Runtime output
xmlFileName:
    takeOffAndLandingWeatherReports.xml
xsdFileName:
    takeOffAndLandingWeatherReports.xsd
takeOffAndLandingWeatherReports.xml is invalid (against takeOffAndLandingWeatherReports.xsd)
(line 14, column 12):
 cvc-complex-type.2.4.a: Invalid content was found starting with element 'fonzi'. One of
'{"http://tempuri.org/TakeOffAndLandingWeatherReports.xsd":wind}' is expected.
Process finished with exit code -1
```

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/jaxp/stax/using.html
C:\Users\John\Documents\Sda\Code\BjaxXsl\Examples\XmlAviationWeather\JaxpDemo\src\stax\StaxXmlIteratorReader.java

Stax Iterator API - Write XML

Take a text file of arbitrary format and output it as XML using Stax (Iterator API/XMLEventWriter):

Setup a scanner for the input file.

```
static public void IteratorApiWriter(String txtFileName,
                                      OutputDirection outputDirection) throws
        UnexpectedException {

    //  Setup a scanner for the input file.
    File inputFile = new File(txtFileName);
    BufferedReader bufferedReader = null;
    Scanner scanner = null;
    try {
        bufferedReader = new BufferedReader(new FileReader(inputFile));
        scanner = new Scanner(bufferedReader);

    } catch (FileNotFoundException e) {
        e.printStackTrace();
    }
```

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/jaxp/stax/using.html
C:\Users\John\Documents\Sda\Code\BjaxXsl\Examples\XmlAviationWeather\JaxpDemo\src\stax\StaxXmlIteratorWriter.java

Create a XMLEventWriter for your desired output stream (e.g. file or screen).

```
    // Create a XMLEventWriter for your desired output stream (e.g. file or screen).
    XMLOutputFactory xmlOutputFactory = XMLOutputFactory.newInstance();
```

```
    xmlOutputFactory.setProperty(XMLOutputFactory.IS_REPAIRING_NAMESPACES, false);
    XMLEventFactory xmlEventFactory = XMLEventFactory.newInstance();
    XMLEventWriter xmlEventWriter = null;
    try {
        switch (outputDirection) {
            case FILE:
                final String outputFileName =
                        "takeOffAndLandingWeatherReportResult" + ".xml";
                try {
                    FileOutputStream fileOutputStream = new FileOutputStream(
                            outputFileName);
                    xmlEventWriter = xmlOutputFactory.createXMLEventWriter(
                            fileOutputStream);
                    System.out.println("outputFileName: " + outputFileName);
                } catch (FileNotFoundException e) {
                    e.printStackTrace();
                }
                break;

            case SCREEN:
                xmlEventWriter = xmlOutputFactory.createXMLEventWriter(System.out);
                break;

            default:
                throw new UnexpectedException("Unexpected outputDirection: " +
                                                    outputDirection.toString());
        }
    } catch (XMLStreamException e) {
        e.printStackTrace();
    }
```

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/jaxp/stax/using.html
C:\Users\John\Documents\Sda\Code\BjaxXsl\Examples\XmlAviationWeather\JaxpDemo\src\stax\StaxXmlIteratorWriter.java

Process the input file using a scanner. In this example we use line by line operations. As you process the input file write to XML using the XML writer.

```
    try {
        String field;
        String value;
        // Find one or more characters that are not "~" and not a space.
        String regexPattern = "[^~\\s]+";
        xmlEventWriter.add(xmlEventFactory.createStartDocument());
        xmlEventWriter.add(xmlEventFactory.createCharacters("\r\n"));
        xmlEventWriter.add(xmlEventFactory.createStartElement("", null, "report"));
        xmlEventWriter.add(xmlEventFactory.createCharacters("\r\n"));
        while (scanner.hasNextLine()) {
            field = scanner.findInLine(regexPattern);
            // findInLine consumes that which it matches. So we can apply the same
            // regex pattern to consume more of the line.
            value = scanner.findInLine(regexPattern);
            scanner.nextLine();
            xmlEventWriter.add(xmlEventFactory.createCharacters("  "));
            if (field.equals("dateTimeUtc")) {
                // Insert value as an attribute
                xmlEventWriter.add(xmlEventFactory.createStartElement("",
                                                                      null,
                                                                      field));
                xmlEventWriter.add(xmlEventFactory.createAttribute("Zulu", value));
                xmlEventWriter.add(xmlEventFactory.createEndElement("", null, field));
            } else {
                // Insert value as an element character
                xmlEventWriter.add(xmlEventFactory.createStartElement("",
                                                                      null,
                                                                      field));
                xmlEventWriter.add(xmlEventFactory.createCharacters(value));
                xmlEventWriter.add(xmlEventFactory.createEndElement("", null, field));
            }
            xmlEventWriter.add(xmlEventFactory.createCharacters("\r\n"));

        } // while

        xmlEventWriter.add(xmlEventFactory.createEndElement("", null, "report"));
```

```
        xmlEventWriter.add(xmlEventFactory.createEndDocument());
        xmlEventWriter.add(xmlEventFactory.createCharacters("\r\n"));
        xmlEventWriter.add(xmlEventFactory.createCharacters("\r\n"));
        xmlEventWriter.flush();

        if (xmlEventWriter != null) {
            xmlEventWriter.close();
            System.out.println("Output success.");
        } // Try
    } catch (XMLStreamException e) {
        e.printStackTrace();
    } finally {
        if (scanner != null) {
            scanner.close();
        }
    }
```

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/jaxp/stax/using.html
C:\Users\John\Documents\Sda\Code\BjaxXsl\Examples\XmlAviationWeather\JaxpDemo\src\stax\StaxXmlIteratorWriter.java

Example input and output.

```
// Input.  takeOffAndLandingWeatherReportValuePairs.txt
location~ YMMLX
dateTimeUtc~ 2003-05-16T16:00:00Z
wind~ 06014
direction~ 060
speed~ 14
qnh~ 1013.2
temperature~ 05
dewPoint~ 02
sky~
cavok~
remarks~ Runway 16L closed for repair.

// Output. takeOffAndLandingWeatherReportResult.xml
<?xml version="1.0"?>
<report>
  <location>YMMLX</location>
  <dateTimeUtc Zulu="2003-05-16T16:00:00Z"></dateTimeUtc>
  <wind>06014</wind>
  <direction>060</direction>
  <speed>14</speed>
  <qnh>1013.2</qnh>
  <temperature>05</temperature>
  <dewPoint>02</dewPoint>
  <sky></sky>
  <cavok></cavok>
  <remarks>Runway</remarks>
</report>
```

Note the empty element is outputted using two tags, not one. This is a limitation when using XMLEventWriter (and XMLEventFactory). If you need an empty element to output only one tag then use the Stax Cursor API.

Networking

Overview

Networking in java involves working with one of:

- URL streams;
- Sockets;
- Datagrams.

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/networking/TOC.html

Those three kinds of working use either TCP or UDP as the underlying internet transport protocol.

- TCP: "a connection-based protocol that provides a reliable flow of data between two computers."
- UDP: "sends independent packets of data, called datagrams, from one computer to another with no guarantees about arrival. UDP is not connection-based like TCP."

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/networking/overview/networking.html

If you are connecting to the web you'll generally want to use URL streams (URL class and related classes (URLConnection, URLEncoder)), rather than sockets or datagrams.

URL streams "are a relatively high-level connection to the Web and use sockets as part of the underlying implementation"

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/networking/sockets/definition.html

The java networking namespace is java.net . Some of the chief classes within include:

- URL, URLConnection, URI (for TCP operations);
- Socket, ServerSocket (for TCP operations);
- DatagramPacket, DatagramSocket, MulticastSocket (for UDP operations).

URIs and URLs

Anatomy of URIs and URLs

A Uniform Resource Identifier (URI) is either a:

- Uniform Resource Name (URN); or
- Uniform Resource Locator (URL).

(Oracle, 2017) http://docs.oracle.com/javase/7/docs/api/java/net/URI.html

In Java the difference between the URI and URL classes include:

- "a URI instance is little more than a structured string that supports the syntactic, scheme-independent operations of comparison, normalization, resolution, and relativization."
- " a URL is a structured string that supports the syntactic operation of resolution as well as the network I/O operations of looking up the host and opening a connection to the specified resource."

(Oracle, 2017) http://docs.oracle.com/javase/7/docs/api/java/net/URI.html

URI Syntax:

- `[scheme:]scheme-specific-part[#fragment]`

```
// A hierarchical URI is subject to further parsing
[scheme:][//authority][path][?query][#fragment]

// A server-based authority of a hierarchial URI is further subject to the following
[user-info@]host[:port]
```

(Oracle, 2017) http://docs.oracle.com/javase/7/docs/api/java/net/URI.html

URL Syntax:

- `[scheme:][//authority][path][?query][#ref]`

```
// Authority breakdown
[user-info@]host[:port]
```

Take a URL like http://john:password@www.philorum.org:80/constitution.html?first_name=John&last_name=Doe#freedomOfSpeech. It has the following parts:

URI Class Name	URL Class Name	Example
scheme	protocol	http
scheme-specficic-part		//john:password@www.philorum.org:80/constitution.html?first_name=John&last_name=Doe
	authority	john:password@www.philorum.org:80
user-info	user-info	john:password
host	host	www.philorum.org
port	port	80
	file	/constitution.html?first_name=John&last_name=Doe
path	path	/constitution.html
query	query	first_name=John&last_name=Doe
fragment	ref	freedomOfSpeech

http://en.wikipedia.org/wiki/Uniform_resource_locator
(Oracle, 2017) http://docs.oracle.com/javase/tutorial/networking/urls/definition.html
(Oracle, 2014) http://docs.oracle.com/javase/7/docs/api/java/net/URI.html
(Oracle, 2017) http://docs.oracle.com/javase/7/docs/api/java/net/URL.html#set%28java.lang.String,%20java.lang.String,%20int,%20java.lang.String,%20java.lang.String,%20java.lang.String,%20java.lang.String,%20java.lang.String%29
C:\Users\John\Documents\Sda\Code\Java\Examples\TutorialAtOracle\HelloWorld\src\helloworld\net\NetworkDemo.java

Create a URL

Create a URL. Basic method.

```
URL myUrl = new URL(
        "http://john:password@www.philorum.org:80/constitution" +
                ".html?first_name=John&last_name=Doe#freedomOfSpeech");
System.out.println(myUrl.toString());

// Output
http://www.philorum.org:80/constitution.html#freedomOfSpeech
```

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/networking/urls/creatingUrls.html
C:\Users\John\Documents\Sda\Code\Java\Examples\TutorialAtOracle\HelloWorld\src\helloworld\net\NetworkDemo.java

Create a URL, one component at a time.

```
URL myUrl02 = new URL("http",
                      "www.philorum.org",
                      80,
                      "/constitution" + ".html#freedomOfSpeech");
System.out.println(myUrl02.toString());

// Output
http://www.philorum.org:80/constitution.html#freedomOfSpeech
```

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/networking/urls/creatingUrls.html
C:\Users\John\Documents\Sda\Code\Java\Examples\TutorialAtOracle\HelloWorld\src\helloworld\net\NetworkDemo.java

Create relative URLS.

```
URL baseUrl = new URL("http://www.philorum.org");
URL relativeUrl = new URL(baseUrl, "constitution.html");
System.out.println(relativeUrl);

// Output
http://www.philorum.org/constitution.html
```

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/networking/urls/creatingUrls.html
C:\Users\John\Documents\Sda\Code\Java\Examples\TutorialAtOracle\HelloWorld\src\helloworld\net\NetworkDemo.java

Create relative URL with reference (to an anchor).

```
URL anchoredURL = new URL(relativeUrl, "#freedomOfSpeech");
System.out.println(anchoredURL);

// Output
http://www.philorum.org/constitution.html#freedomOfSpeech
```

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/networking/urls/creatingUrls.html
C:\Users\John\Documents\Sda\Code\Java\Examples\TutorialAtOracle\HelloWorld\src\helloworld\net\NetworkDemo.java

Create a URL that handles that handles special characters (by encoding them).

```
URI myUri = new URI("http",
                    "john:password@www.philorum.org:80",
                    "/cool page.html",
                    "first_name=John&last_name=Doe",
                    "freedomOfSpeech");
URL myURL = myUri.toURL();
```

```
System.out.println(myURL.toString());

// Output
http://john:password@www.philorum.org:80/cool%20page.html?first_name=John&last_name=Doe#free
domOfSpeech
```

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/networking/urls/creatingUrls.html
C:\Users\John\Documents\Sda\Code\Java\Examples\TutorialAtOracle\HelloWorld\src\helloworld\net\NetworkDemo.java

Output Parts of a URI or URL

Output parts of a URI or URL ("parsing a URL or URI").

```
// Create a URL
URL myUrl = new URL("http://john:password@www.philorum.org:80/constitution"
                            + ".html?first_name=John&last_name=Doe#freedomOfSpeech");
System.out.println(myUrl.toString());

outputUrlParts(myUrl);
outputUriParts(myUrl.toURI());

static public void outputUrlParts(URL url) {
    System.out.println();
    System.out.println("URL Parts");
    System.out.println("Protocol: " + url.getProtocol());
    System.out.println("Authority: " + url.getAuthority());
    System.out.println("User Info: " + url.getUserInfo());
    System.out.println("Host: " + url.getHost());
    System.out.println("Port: " + url.getPort());
    System.out.println("File: " + url.getFile());
    System.out.println("Path: " + url.getPath());
    System.out.println("Query: " + url.getQuery());
    System.out.println("Ref: " + url.getRef());
}

static public void outputUriParts(URI uri) {
    System.out.println();
    System.out.println("URI Parts");
    System.out.println("Scheme: " + uri.getScheme());
    System.out.println("SchemeSpecficPart: " + uri.getSchemeSpecificPart());
    System.out.println("Authority: " + uri.getAuthority());
    System.out.println("User Info: " + uri.getUserInfo());
    System.out.println("Host: " + uri.getHost());
    System.out.println("Port: " + uri.getPort());
    System.out.println("Path: " + uri.getPath());
    System.out.println("Query: " + uri.getQuery());
    System.out.println("Fragment: " + uri.getFragment());
}
```

For results see Anatomy of URIs and URLs above.

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/networking/urls/urlInfo.html
C:\Users\John\Documents\Sda\Code\Java\Examples\TutorialAtOracle\HelloWorld\src\helloworld\net\NetworkDemo.java

Read/Write Network content

Read Content from a URL directly - Server: static content.

To read the content (e.g. a web page) at a given URL, from the URL directly ...

```
static public void start() {
    readContentAtUrlFromUrlDirectly(
            "http://localhost:8080/web/Libraries/XhtmlLibrary/BasicHtml5.html");
    readContentAtUrlFromUrlDirectly("http://www.oracle.com");
```

```
}

static public void readContentAtUrlFromUrlDirectly(String urlString) {
    try {
        // Create a URL
        URL url = new URL(urlString);

        // Create an InputStreamReader based up the InputStream returned by
        // url.openStream
        BufferedReader reader = new BufferedReader(
                new InputStreamReader(url.openStream()));
        String readLine;
        while ((readLine = reader.readLine()) != null) {
            System.out.println(readLine);
        }
        reader.close();

    } catch (MalformedURLException e) {
        e.printStackTrace();
    } catch (IOException e) {
        e.printStackTrace();
    }
}

// Output (outpus entire web page)
!DOCTYPE html>
<html lang="en-GB">
<head>
  <title>Basic HTML</title>
...
```

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/networking/urls/readingURL.html
C:\Users\John\Documents\Sda\Code\Java\Examples\TutorialAtOracle\HelloWorld\src\helloworld\net\NetworkDemo.java

Reading from a URL using a URLConnection might be more useful than reading directly from a URL.

A URLConnection provides more flexibility. At the very least it allows you to also write to a URL.

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/networking/urls/readingWriting.html

Read from a URL, using a HttpURLConnection (or URLConnection). This produces the same output as from readContentAtUrlFromUrlDirectly().

```
static public void start() {
readContentAtUrlWithConnection("http://localhost:8080/web/Libraries/XhtmlLibrary/BasicHtml5.
html");
}

static public void readContentAtUrlWithConnection(String urlString) {
    java.net.HttpURLConnection urlConnection = null;

    // Alternatively we could use URLConnection
    // java.net.URLConnection urlConnection = null;
    try {
        URL url = new java.net.URL(urlString);
        urlConnection = (java.net.HttpURLConnection) url.openConnection();

        // Alternatively we could use URLConnection
        // urlConnection = url.openConnection();

        urlConnection.connect();

        // Create an InputStreamReader based up the InputStream returned by
        // urlConnection.getInputStream() (this implicitly calls the connect
        // method, in case one hadn't been explicitly called).
        BufferedReader reader = new BufferedReader(new InputStreamReader
                                                    (urlConnection
                                                            .getInputStream
                                                                    ()));
```

```
        String readLine;
        while ((readLine = reader.readLine()) != null) {
            System.out.println(readLine);
        }
        reader.close();

    } catch (MalformedURLException e) {
        e.printStackTrace();
    } catch (IOException e) {
        e.printStackTrace();
    } finally {

        // Comment this out if using a plane URLConnection rather than
        // HttpURLConnection
        urlConnection.disconnect();
    }
}


// Output (outpus entire web page)
!DOCTYPE html>
<html lang="en-GB">
<head>
  <title>Basic HTML</title>
...
```

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/networking/urls/connecting.html
(Oracle, 2017) http://docs.oracle.com/javase/tutorial/networking/urls/readingWriting.html
C:\Users\John\Documents\Sda\Code\Java\Examples\TutorialAtOracle\HelloWorld\src\helloworld\net\NetworkDemo.java

Write to and read from a URL

Server: Php dynamic content. Using URLConnection (could use HttpUrlConnection).

Writing to a URL, using a URLConnection. In this example we write (POST) to a web server and read back the responding web page.

```
static public void start() {
    WriteToAndReadFromAUrl.serverPhp();
}

/**
 * This will effect a POST to the server, from which a responding web page is
 * received and output to the screen.
 * <p/>
 * That a POST has occurred was verified by examining Apache access.log
 */
static public void serverPhp() {
    String urlString =
            "http://localhost:8080/Php/Examples/formMultifieldHandleJustDisplay" +
            "" + ".php";
    URL url = null;
    try {
        url = new URL(urlString);
        // Could use java.net.HttpUrlConnection instead.
        URLConnection urlConnection = url.openConnection();
        urlConnection.setDoOutput(true);

        OutputStreamWriter writer = new OutputStreamWriter(urlConnection
                                                           .getOutputStream());

        writer.write("cboAreaID=1&");
        writer.write("txtLaunchName=Coronet&");
        writer.write("txtAltitude=5,000ft&");
        writer.write("txtWindDirection=NE&");
        writer.write("txtRating=Novice");
        writer.close();

        BufferedReader reader = new BufferedReader(new InputStreamReader(
                urlConnection.getInputStream()));
```

```
        String readLine;
        while ((readLine = reader.readLine()) != null) {
            System.out.println(readLine);
        }
        reader.close();
    } catch (MalformedURLException e) {
        e.printStackTrace();
    } catch (IOException e) {
        e.printStackTrace();
    }
}

// Responding web page written to the screen
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.1//EN"
      "http://www.w3.org/TR/xhtml11/DTD/xhtml11.dtd">
...
<html xmlns="http://www.w3.org/1999/xhtml" xml:lang="en">
<head>
  ...
</head>
<body>
...
<div>
<pre>
LaunchName: Coronet
Altitude: 5,000ft
Wind Direction: NE
Rating:  Novice
</pre>
...
</body>
</html>
```

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/networking/urls/readingWriting.html
(Bentley, TutorialAtOracle Code Examples, 2013)
C:\Users\John\Documents\Sda\Code\Java\Examples\TutorialAtOracle\HelloWorld\src\helloworld\net\NetworkDemo.java

Server: Java EE Web service, restful. Using HttpURLConnection (or URLConnection).

Coding:

1. Server: Setup a Java EE Web service, restful. See (Bentley, Java Reference - Java EE, 2016), Hello Worlds, Webservice, RESTFull.
2. Client:

```
static public void start() {
    WriteToAndReadFromAUrl.serverJavaEEWebServiceRestful();
}

static class WriteToAndReadFromAUrl {
    /**
     * This will effect a POST to the server, from which a response is received.
     * In this case the Java EE Appserver, served from Glassfish, returns a plain
     * text mime type.
     */
    static public void serverJavaEEWebServiceRestful() {
        // Assumes a JavaEE WebService, Restful setup and deployed at the
        // url.
        String urlString = "http://localhost:8090/HelloWorldWebServiceRestfulworld"
                + "/HelloWorldWebServiceRestfulworld";

        URL url = null;
        try {
            url = new URL(urlString);
            // Could use java.net.HttpUrlConnection instead.
            URLConnection urlConnection = url.openConnection();
            urlConnection.setDoOutput(true);

            OutputStreamWriter writer = new OutputStreamWriter(urlConnection
                                                               .getOutputStream());
```

```
            writer.write("txtName=Sergio");
            writer.close();

            BufferedReader reader = new BufferedReader(new InputStreamReader(
                    urlConnection.getInputStream()));
            String readLine;
            while ((readLine = reader.readLine()) != null) {
                System.out.println(readLine);
            }
            reader.close();
        } catch (MalformedURLException e) {
            e.printStackTrace();
        } catch (IOException e) {
            e.printStackTrace();
        }
    }
}

// Output
Hello Sergio
```

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/networking/urls/readingWriting.html
(Bentley, TutorialAtOracle Code Examples, 2013)
C:\Users\John\Documents\Sda\Code\Java\Examples\TutorialAtOracle\HelloWorld\src\helloworld\net\NetworkDemo.java

Sockets

Overview

> If you are trying to connect to the Web, the `URL` class and related classes (`URLConnection`, `URLEncoder`) are probably more appropriate than the socket classes. In fact, URLs are a relatively high-level connection to the Web and use sockets as part of the underlying implementation.

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/networking/sockets/definition.html

> URLs and URLConnections provide a relatively high-level mechanism for accessing resources on the Internet. Sometimes your programs require lower-level network communication, for example, when you want to write a client-server application.

(Oracle, 2017) https://docs.oracle.com/javase/tutorial/networking/sockets/index.html

Sockets use TCP.

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/networking/sockets/index.html

> A socket is one endpoint of a two-way communication link between two programs running on the network. ... An endpoint is a combination of an IP address and a port number. Every TCP connection can be uniquely identified by its two endpoints. That way you can have multiple connections between your host and the server. ... A socket [client or server side] is bound to a port number.

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/networking/sockets/definition.html

The java.net package provides two classes, java.net.Socket and java.net.ServerSocket--that implement the client side and server side of the connection, respectively.

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/networking/sockets/index.html

Write to and read from a Socket (client side)

Basic example

In IntelliJ Build EchoClient and EchoServer as artifacts. We then run the resulting jar files from the command line, in separate instances. Observe network connection in Windows Resource Monitor:

1. EchoServer.

```
import java.io.*;
import java.net.*;

public class EchoServer {
    public static void main(String[] args) throws IOException {

        if (args.length != 1) {
            System.err.println("Usage: java EchoServer <port number>");
            System.exit(1);
        }

        int portNumber = Integer.parseInt(args[0]);
        System.out.println("Started Echoserver on port " + portNumber);
        System.out.println("Windows Key > type 'Resource Monitor' > Listening Ports: Click
to expand.");

        try (
                ServerSocket serverSocket =
                        new ServerSocket(portNumber);
                Socket clientSocket = serverSocket.accept();
                PrintWriter out =
                        new PrintWriter(clientSocket.getOutputStream(), true);
                BufferedReader in = new BufferedReader(
                        new InputStreamReader(clientSocket.getInputStream()));
        ) {

            String inputLine;
            while ((inputLine = in.readLine()) != null) {
                out.println("Server: " + inputLine);
            }
        } catch (IOException e) {
            System.out.println("Exception caught when trying to listen on port "
                    + portNumber + " or listening for a connection");
            System.out.println(e.getMessage());
        }
    }
}
```

(Oracle, 2017) https://docs.oracle.com/javase/tutorial/networking/sockets/readingWriting.html
(Bentley, TutorialAtOracle Code Examples, 2013), EchoServer.

2. Compile EchoSever into a Jar file. See (Bentley, Java Reference - Language.docx, 2017) > "Run console applications created by IntelliJ IDEA" > "From a command Line" > "By reference to jar file".
3. Use powershell to start the EchoServer.

```
PS
C:\Users\John\Documents\Sda\Code\Java\Examples\TutorialAtOracle\out\artifacts\EchoServer_jar
> java -jar .\EchoServer.jar 5
```

4. Open Windows "Resource Monitor" to observe the socket connections. Windows Key > type 'Resource Monitor' > Listening Ports: Click to expand.
5. EchoClient

```
import java.io.*;
import java.net.*;
```

```
public class EchoClient {
    public static void main(String[] args) throws IOException {

        if (args.length != 2) {
            System.err.println(
                    "Usage: java EchoClient <host name> <port number>");
            System.exit(1);
        }

        String hostName = args[0];
        int portNumber = Integer.parseInt(args[1]);

        try (
                Socket echoSocket = new Socket(hostName, portNumber);
                PrintWriter out =
                        new PrintWriter(echoSocket.getOutputStream(), true);
                BufferedReader in =
                        new BufferedReader(
                                new InputStreamReader(echoSocket.getInputStream()));
                BufferedReader stdIn =
                        new BufferedReader(
                                new InputStreamReader(System.in))
        ) {
            String userInput;
            while ((userInput = stdIn.readLine()) != null) {
                out.println(userInput);
                // in.readLine waits until info returned from the server.
                System.out.println("Client: " + in.readLine());
            }
        } catch (UnknownHostException e) {
            System.err.println("Don't know about host " + hostName);
            System.exit(1);
        } catch (IOException e) {
            System.err.println("Couldn't get I/O for the connection to " +
                    hostName);
            System.exit(1);
        }
    }
}
```

(Oracle, 2017) https://docs.oracle.com/javase/tutorial/networking/sockets/readingWriting.html
(Bentley, TutorialAtOracle Code Examples, 2013), EchoClient.

6. Compile EchoClient into a Jar file. See (Bentley, Java Reference - Language.docx, 2017) > "Run console applications created by IntelliJ IDEA" > "From a command Line" > "By reference to jar file".
7. Use a seperate powershell instance (in addition to the previously ran and currently alive EchoServer) to start the EchoClient.

```
PS
C:\Users\John\Documents\Sda\Code\Java\Examples\TutorialAtOracle\out\artifacts\EchoClient_jar
> java -jar .\EchoClient.jar localhost 5
```

8. In the powershell EchoClient Instance type some input, hit [Enter], and observe the echo'd output.
9. In Resource Monitor, Listening Ports Section, observe the jave.exe image with the relevant port (5 in this example). In "Process with Network Activity" select the relevant java.exe to filter out network activity we are not interested in (match on the PID).

"Sreams connected to a socket should be closed before the socket itself is closed". This happens automatically in EchoClient given the ordering of statements opened within the `try-with-resources` statement as closing occurs in reverse order.

(Oracle, 2017), " Reading from and Writing to a Socket",
https://docs.oracle.com/javase/tutorial/networking/sockets/readingWriting.html

> This client program is straightforward and simple because the echo server implements a simple protocol. The client sends text to the server, and the server echoes it back. When your client programs are talking to a more complicated server such as an HTTP server, your client program will also be more complicated. However, **the basics are much the same** as they are in this program:
>
> 1. Open a socket.
> 2. Open an input stream and output stream to the socket.
> 3. Read from and write to the stream according to the server's protocol.
> 4. Close the streams.
> 5. Close the socket.
>
> Only step 3 differs from client to client, depending on the server. The other steps remain largely the same. [Emphasis added].

(Oracle, 2017), "Reading from and Writing to a Socket", https://docs.oracle.com/javase/tutorial/networking/sockets/readingWriting.html

Example supporting multiple clients and a custom protocol

> All client/server pairs must have some protocol by which they speak to each other; otherwise, the data that passes back and forth would be meaningless. The protocol that your own clients and servers use depends entirely on the communication required by them to accomplish the task.

(Oracle, 2017), " Writing the Server Side of a Socket", https://docs.oracle.com/javase/tutorial/networking/sockets/clientServer.html

In the KnockKnock Example the Protocols is implemented on the server side (and it contains error messages to remind the client user of the protocol if they provide the wrong input.

(Oracle, 2017), " Writing the Server Side of a Socket", https://docs.oracle.com/javase/tutorial/networking/sockets/clientServer.html
C:\Users\John\Documents\Sda\Code\Java\Examples\TutorialAtOracle\KnockKnockServer\src\KnockKnockServer.java
C:\Users\John\Documents\Sda\Code\Java\Examples\TutorialAtOracle\KnockKnockServer\src\KnockKnockProtocol.java
C:\Users\John\Documents\Sda\Code\Java\Examples\TutorialAtOracle\KnockKnockClient\src\KnockKnockClient.java

You must code your client and sever according to who speaks first.

In the KnockKnock example "The server speaks first, so the client must listen first."

(Oracle, 2017), "Writing the Server Side of a Socket", https://docs.oracle.com/javase/tutorial/networking/sockets/clientServer.html

You can support multiple clients by opening sockets in their own thread. `KnockKnockMultiServer` listens for client connections and opens threads. `KnockKnockMultiServerRunnable` is a Runnable KnockKnockServer class and therefore supports threading.

```
import java.net.*;
import java.io.*;

public class KnockKnockMultiServer {
    public static void main(String[] args) throws IOException {
```

```
        if (args.length != 1) {
            System.err.println("Usage: java KnockKnockMultiServer <port number>");
            System.exit(1);
        }

        int portNumber = Integer.parseInt(args[0]);
        boolean listening = true;

        System.out.println("Waiting for client connection ...");
        try (ServerSocket serverSocket = new ServerSocket(portNumber)) {
            while (listening) {
                Thread t = new Thread(new
KnockKnockMultiServerRunnable(serverSocket.accept()));
                t.start();
            }
        } catch (IOException e) {
            System.err.println("Could not listen on port " + portNumber);
            System.exit(-1);
        }
    }
}
```

(Oracle, 2017), "Writing the Server Side of a Socket", https://docs.oracle.com/javase/tutorial/networking/sockets/clientServer.html
C:\Users\John\Documents\Sda\Code\Java\Examples\TutorialAtOracle\KnockKnockMultiServer\src\KnockKnockMultiServer.java

```
import java.net.*;
import java.io.*;

public class KnockKnockMultiServerRunnable implements Runnable {
    private Socket socket = null;

    public KnockKnockMultiServerRunnable(Socket socket) {
        this.socket = socket;
    }

    public void run() {

        try (
                PrintWriter out = new PrintWriter(socket.getOutputStream(), true);
                BufferedReader in = new BufferedReader(
                        new InputStreamReader(
                                socket.getInputStream()));
        ) {
            String inputLine, outputLine;
            KnockKnockProtocol kkp = new KnockKnockProtocol();

            // The server speaks first with "Knock, Knock"
            outputLine = kkp.processInput(null);
            out.println(outputLine);

            while ((inputLine = in.readLine()) != null) {
                outputLine = kkp.processInput(inputLine);
                out.println(outputLine);
                if (outputLine.equals("Bye"))
                    break;
            }
            socket.close();
        } catch (IOException e) {
            e.printStackTrace();
        }
    }
}
```

(Oracle, 2017), "Writing the Server Side of a Socket", https://docs.oracle.com/javase/tutorial/networking/sockets/clientServer.html

C:\Users\John\Documents\Sda\Code\Java\Examples\TutorialAtOracle\KnockKnockMultiServer\src\KnockKnockMultiServerRunnable.java

The KnockKnockProtocol Class abstracts the protocol away from the socket creating and processing classes.

```
public class KnockKnockProtocol {
    private static final int WAITING = 0;
    private static final int SENTKNOCKKNOCK = 1;
    private static final int SENTCLUE = 2;
    private static final int ANOTHER = 3;

    private static final int NUMJOKES = 5;

    private int state = WAITING;
    private int currentJoke = 0;

    private String[] clues = { "Turnip", "Little Old Lady", "Atch", "Who", "Who" };
    private String[] answers = { "Turnip the heat, it's cold in here!",
            "I didn't know you could yodel!",
            "Bless you!",
            "Is there an owl in here?",
            "Is there an echo in here?" };

    public String processInput(String theInput) {
        String theOutput = null;

        if (state == WAITING) {
            theOutput = "Knock! Knock!";
            state = SENTKNOCKKNOCK;
        } else if (state == SENTKNOCKKNOCK) {
            if (theInput.equalsIgnoreCase("Who's there?")) {
                theOutput = clues[currentJoke];
                state = SENTCLUE;
            } else {
                theOutput = "You're supposed to say \"Who's there?\"! " +
                        "Try again. Knock! Knock!";
            }
        } else if (state == SENTCLUE) {
            if (theInput.equalsIgnoreCase(clues[currentJoke] + " who?")) {
                theOutput = answers[currentJoke] + " Want another? (y/n)";
                state = ANOTHER;
            } else {
                theOutput = "You're supposed to say \"" +
                        clues[currentJoke] +
                        " who?\"" +
                        "! Try again. Knock! Knock!";
                state = SENTKNOCKKNOCK;
            }
        } else if (state == ANOTHER) {
            if (theInput.equalsIgnoreCase("y")) {
                theOutput = "Knock! Knock!";
                if (currentJoke == (NUMJOKES - 1))
                    currentJoke = 0;
                else
                    currentJoke++;
                state = SENTKNOCKKNOCK;
            } else {
                theOutput = "Bye.";
                state = WAITING;
            }
        }
        return theOutput;
    }
}
```

(Oracle, 2017), "Writing the Server Side of a Socket",
https://docs.oracle.com/javase/tutorial/networking/sockets/clientServer.html
C:\Users\John\Documents\Sda\Code\Java\Examples\TutorialAtOracle\KnockKnockMultiServer\src\KnockKnockProtocol.java

The KnockKnockClient

```
import java.io.*;
import java.net.*;
```

```
public class KnockKnockClient {
    public static void main(String[] args) throws IOException {

        if (args.length != 2) {
            System.err.println(
                    "Usage: java EchoClient <host name> <port number>");
            System.exit(1);
        }

        String hostName = args[0];
        int portNumber = Integer.parseInt(args[1]);

        try (
                Socket kkSocket = new Socket(hostName, portNumber);
                PrintWriter out = new PrintWriter(kkSocket.getOutputStream(), true);
                BufferedReader in = new BufferedReader(
                        new InputStreamReader(kkSocket.getInputStream()));
        ) {
            BufferedReader stdIn =
                    new BufferedReader(new InputStreamReader(System.in));
            String fromServer;
            String fromUser;

            while ((fromServer = in.readLine()) != null) {
                System.out.println("Server: " + fromServer);
                if (fromServer.equals("Bye."))
                    break;

                fromUser = stdIn.readLine();
                if (fromUser != null) {
                    System.out.println("Client: " + fromUser);
                    out.println(fromUser);
                }
            }
        } catch (UnknownHostException e) {
            System.err.println("Don't know about host " + hostName);
            System.exit(1);
        } catch (IOException e) {
            System.err.println("Couldn't get I/O for the connection to " +
                    hostName);
            System.exit(1);
        }
    }
}
```

(Oracle, 2017), "Writing the Server Side of a Socket", https://docs.oracle.com/javase/tutorial/networking/sockets/clientServer.html
C:\Users\John\Documents\Sda\Code\Java\Examples\TutorialAtOracle\KnockKnockClient\src\KnockKnockClient.java

Datagrams

Intro

> [As opposed to ...] the reliable, point-to-point channel provided by TCP ...
> a datagram is an independent, self-contained message sent over the network whose arrival, [order of arrival], arrival time, and content are not guaranteed.

(Oracle, 2017), " Lesson: All About Datagrams", https://docs.oracle.com/javase/tutorial/networking/datagrams/index.html

Datagrams use UDP rather than TCP.

(Oracle, 2017), " Lesson: All About Datagrams", https://docs.oracle.com/javase/tutorial/networking/datagrams/index.html

> The `java.net` package contains three classes to help you write Java programs that use datagrams to send and receive packets over the network: `DatagramSocket`, `DatagramPacket`, and `MulticastSocket`. An application can send and receive `DatagramPackets` through a `DatagramSocket`. In addition, `DatagramPackets` can be broadcast to multiple recipients all listening to a `MulticastSocket`.

(Oracle, 2017), "What Is a Datagram?", https://docs.oracle.com/javase/tutorial/networking/datagrams/definition.html

Serve info upon client request. Multiple clients. Server increments state regardless of the number of clients.

Here we have an example where:

- The server sends out info to a client: upon client request.
- The server can handle multiple clients.
- But the server increments information (in our example, advances to the next quote) regardless of the number of clients. (That is, in our example the clients will get a different set of quotes).

See above - Sockets, "Write to and read from a Socket (Client Side)", Basic example - for details on building *.jar files via the creation of IntelliJ "artifacts".

Create a server (console) app that can handle multiple requests from clients (but increment state regardless of the number of clients):

```
import java.io.*;
import java.net.*;
import java.util.*;

public class QuoteServerThread extends Thread {

    protected DatagramSocket socket = null;
    protected BufferedReader in = null;
    protected boolean moreQuotes = true;

    public QuoteServerThread() throws IOException {
        this("QuoteServerThread");
    }

    public QuoteServerThread(String name) throws IOException {
        super(name);
        socket = new DatagramSocket(4445);

        try {
            in = new BufferedReader(new FileReader("one-liners.txt"));
            System.out.println("Waiting for clients ...");
        } catch (FileNotFoundException e) {
            System.err.println("Could not open quote file. Serving time instead.");
        }
    }

    public void run() {

        System.out.println("Each dot will represent a quote sent to client:");
        while (moreQuotes) {
            try {
                byte[] buf = new byte[256];

                // receive request
                DatagramPacket packet = new DatagramPacket(buf, buf.length);
                socket.receive(packet);

                // figure out response
                String dString = null;
                if (in == null)
                    dString = new Date().toString();
```

```
                else
                    dString = getNextQuote();

                buf = dString.getBytes();

                // send the response to the client at "address" and "port"
                InetAddress address = packet.getAddress();
                int port = packet.getPort();
                packet = new DatagramPacket(buf, buf.length, address, port);
                socket.send(packet);
                System.out.print(".");
            } catch (IOException e) {
                e.printStackTrace();
                moreQuotes = false;
            }
        }
        socket.close();
    }

    protected String getNextQuote() {
        String returnValue = null;
        try {
            if ((returnValue = in.readLine()) == null) {
                in.close();
                moreQuotes = false;
                returnValue = "No more quotes. Goodbye.";
            }
        } catch (IOException e) {
            returnValue = "IOException occurred in server.";
        }
        return returnValue;
    }
}


import java.io.*;

public class QuoteServer {
    public static void main(String[] args) throws IOException {
        new QuoteServerThread().start();
    }
}
```

(Oracle, 2017), " Writing a Datagram Client and Server",
https://docs.oracle.com/javase/tutorial/networking/datagrams/clientServer.html
C:\Users\John\Documents\Sda\Code\Java\Examples\TutorialAtOracle\QuoteServer\src\QuoteServer.java
C:\Users\John\Documents\Sda\Code\Java\Examples\TutorialAtOracle\QuoteServer\src\QuoteServerThread.java

Creat the client:

```
import java.io.*;
import java.net.*;

public class QuoteClient {
    public static void main(String[] args) throws IOException {

        if (args.length != 1) {
            System.out.println("Usage: java QuoteClient <hostname>");
            return;
        }

        // get a datagram socket
        DatagramSocket socket = new DatagramSocket();

        // send request
        byte[] buf = new byte[256];
        InetAddress address = InetAddress.getByName(args[0]);
        DatagramPacket packet = new DatagramPacket(buf, buf.length, address, 4445);
        socket.send(packet);

        // get response
        packet = new DatagramPacket(buf, buf.length);
        socket.receive(packet);
```

```
        // display response
        String received = new String(packet.getData(), 0, packet.getLength());
        System.out.println("Quote of the Moment: " + received);

        socket.close();
    }
}
```

(Oracle, 2017), " Writing a Datagram Client and Server",
https://docs.oracle.com/javase/tutorial/networking/datagrams/clientServer.html
C:\Users\John\Documents\Sda\Code\Java\Examples\TutorialAtOracle\QuoteClient\src\QuoteClient.java

Compile QuoteClient and QuoteServer modules into a Jar files. See (Bentley, Java Reference - Language.docx, 2017) > "Run console applications created by IntelliJ IDEA" > "From a command Line" > "By reference to jar file".

Put `one-liners.txt` into the same directory as the QuoteServer.jar.

```
Life is wonderful. Without it we'd all be dead.
Daddy, why doesn't this magnet pick up this floppy disk?
Give me ambiguity or give me something else.
I.R.S.: We've got what it takes to take what you've got!
We are born naked, wet and hungry. Then things get worse.
Make it idiot proof and someone will make a better idiot.
He who laughs last thinks slowest!
Always remember you're unique, just like everyone else.
"More hay, Trigger?" "No thanks, Roy, I'm stuffed!"
A flashlight is a case for holding dead batteries.
...
```

Run QuoteServer then 2 QuoteClient jars in powersell.

```
// Server Instance
PS C:\Users\John> sl
C:\Users\John\Documents\Sda\Code\Java\Examples\TutorialAtOracle\out\artifacts\QuoteServer_ja
r\
PS
C:\Users\John\Documents\Sda\Code\Java\Examples\TutorialAtOracle\out\artifacts\QuoteServer_ja
r> java -jar .\QuoteServer.jar


// Client Instance
PS C:\Users\John> sl
C:\Users\John\Documents\Sda\Code\Java\Examples\TutorialAtOracle\out\artifacts\QuoteClient_ja
r\
PS
C:\Users\John\Documents\Sda\Code\Java\Examples\TutorialAtOracle\out\artifacts\QuoteClient_ja
r> java -jar .\QuoteClien
t.jar localhost
Quote of the Moment: Life is wonderful. Without it we'd all be dead.
PS
C:\Users\John\Documents\Sda\Code\Java\Examples\TutorialAtOracle\out\artifacts\QuoteClient_ja
r> java -jar .\QuoteClien
t.jar localhost
Quote of the Moment: Daddy, why doesn't this magnet pick up this floppy disk?
PS
C:\Users\John\Documents\Sda\Code\Java\Examples\TutorialAtOracle\out\artifacts\QuoteClient_ja
r>
```

C:\Users\John\Documents\Sda\Code\Java\Examples\TutorialAtOracle\out\artifacts\QuoteServer_jar\QuoteServer.jar
C:\Users\John\Documents\Sda\Code\Java\Examples\TutorialAtOracle\out\artifacts\QuoteServer_jar\one-liners.txt
C:\Users\John\Documents\Sda\Code\Java\Examples\TutorialAtOracle\out\artifacts\QuoteClient_jar\QuoteClient.jar

Repeatedly execute the client commands in the two client windows that are open. Observe that the next quote from `one-liners.txt` gets feed so that no client has a quote that the other has. In other words, the server increments state regardless of the number of clients.

Serve info as a broadcast to listening (multiple) clients.

The server broadcasts a message (a quote in this example) and multiple clients receive the same message.

See above - Sockets, "Write to and read from a Socket (Client Side)", Basic example - for details on building *.jar files via the creation of IntelliJ "artifacts".

The Server

```
import java.io.BufferedReader;
import java.io.FileNotFoundException;
import java.io.FileReader;
import java.io.IOException;
import java.net.DatagramPacket;
import java.net.DatagramSocket;
import java.net.InetAddress;
import java.util.Date;

public class QuoteServerThread extends Thread {

    protected DatagramSocket socket = null;
    protected BufferedReader in = null;
    protected boolean moreQuotes = true;

    public QuoteServerThread() throws IOException {
        this("QuoteServerThread");
    }

    public QuoteServerThread(String name) throws IOException {
        super(name);
        socket = new DatagramSocket(4445);

        try {
            in = new BufferedReader(new FileReader("one-liners.txt"));
        } catch (FileNotFoundException e) {
            System.err.println("Could not open quote file. Serving time instead.");
        }
    }

    public void run() {

        System.out.println("Each dot will represent a quote sent to client:");
        while (moreQuotes) {
            try {
                byte[] buf = new byte[256];

                // receive request
                DatagramPacket packet = new DatagramPacket(buf, buf.length);
                socket.receive(packet);

                // figure out response
                String dString = null;
                if (in == null)
                    dString = new Date().toString();
                else
                    dString = getNextQuote();

                buf = dString.getBytes();

                // send the response to the client at "address" and "port"
                InetAddress address = packet.getAddress();
                int port = packet.getPort();
                packet = new DatagramPacket(buf, buf.length, address, port);
                socket.send(packet);
```

```
                System.out.print(".");
            } catch (IOException e) {
                e.printStackTrace();
                moreQuotes = false;
            }
        }
        socket.close();
    }

    protected String getNextQuote() {
        String returnValue = null;
        try {
            if ((returnValue = in.readLine()) == null) {
                in.close();
                moreQuotes = false;
                returnValue = "No more quotes. Goodbye.";
            }
        } catch (IOException e) {
            returnValue = "IOException occurred in server.";
        }
        return returnValue;
    }
}


import java.io.*;
import java.net.*;
import java.util.*;

public class QuoteServerMulticastThread extends QuoteServerThread {

    private long FIVE SECONDS = 5000;

    public QuoteServerMulticastThread() throws IOException {
        super("QuoteServerMulticastThread");
        System.out.println("Broadcasting to clients ...");
    }

    @Override
    public void run() {
        while (moreQuotes) {
            try {
                byte[] buf = new byte[256];

                // construct quote
                String dString = null;
                if (in == null)
                    dString = new Date().toString();
                else
                    dString = getNextQuote();
                buf = dString.getBytes();

                // send it
                InetAddress group = InetAddress.getByName("230.0.0.1");
                DatagramPacket packet = new DatagramPacket(buf, buf.length, group, 4446);
                socket.send(packet);
                System.out.println("Sent: " + dString);

                // sleep for a while
                try {
                    sleep((long)(Math.random() * FIVE_SECONDS));
                } catch (InterruptedException e) { }
            } catch (IOException e) {
                e.printStackTrace();
                moreQuotes = false;
            }
        }
        socket.close();
    }
}

public class QuoteServerMulticast {
    public static void main(String[] args) throws java.io.IOException {
        new QuoteServerMulticastThread().start();
    }
}
```

(Oracle, 2017) https://docs.oracle.com/javase/tutorial/networking/datagrams/broadcasting.html
C:\Users\John\Documents\Sda\Code\Java\Examples\TutorialAtOracle\QuoteServerMulticast

The Client:

```
import java.io.IOException;
import java.net.DatagramPacket;
import java.net.InetAddress;
import java.net.MulticastSocket;

public class QuoteClientMulticast {

    public static void main(String[] args) throws IOException {

        MulticastSocket socket = new MulticastSocket(4446);
        InetAddress address = InetAddress.getByName("230.0.0.1");
        socket.joinGroup(address);

        DatagramPacket packet;

        // get a few quotes
        for (int i = 0; i < 15; i++) {

            byte[] buf = new byte[256];
            packet = new DatagramPacket(buf, buf.length);
            socket.receive(packet);

            String received = new String(packet.getData(), 0, packet.getLength());
            System.out.println("Quote of the Moment: " + received);
        }

        socket.leaveGroup(address);
        socket.close();
    }
}
```

(Oracle, 2017) https://docs.oracle.com/javase/tutorial/networking/datagrams/broadcasting.html
C:\Users\John\Documents\Sda\Code\Java\Examples\TutorialAtOracle\QuoteClientMulticast

Network Parameters

Intro

Sometimes you might want to access the paramters of active network connections (e.g. via a wired Ethernet, wireless 802.11n, wireless Bluetooth). This is achived via `java.net.NetworkInterface`.

(Oracle, 2017) https://docs.oracle.com/javase/tutorial/networking/nifs/index.html

List Network Interfaces ("Network Adapters") and list their parameters.

```
import java.io.PrintStream;
import java.net.InetAddress;
import java.net.NetworkInterface;
import java.net.SocketException;
import java.util.Arrays;
import java.util.Collections;
import java.util.Enumeration;

public class NetworkInterfaceDemo {

    static PrintStream out = System.out;

    public static void start() throws SocketException {
        ListNetworkInterfaces();
    }
```

```
    static void ListNetworkInterfaces () throws SocketException {
        Enumeration<NetworkInterface> nets = NetworkInterface.getNetworkInterfaces();
        for (NetworkInterface netint : Collections.list(nets))
            displayInterfaceInformation(netint);
    }
    static void displayInterfaceInformation(NetworkInterface netint) throws SocketException
{
        out.printf("Display name: %s\n", netint.getDisplayName());
        out.printf("Name: %s\n", netint.getName());
        Enumeration<InetAddress> inetAddresses = netint.getInetAddresses();

        for (InetAddress inetAddress : Collections.list(inetAddresses)) {
            out.printf("InetAddress: %s\n", inetAddress);
        }

        out.printf("Up? %s\n", netint.isUp());
        out.printf("Loopback? %s\n", netint.isLoopback());
        out.printf("PointToPoint? %s\n", netint.isPointToPoint());
        out.printf("Supports multicast? %s\n", netint.supportsMulticast());
        out.printf("Virtual? %s\n", netint.isVirtual());
        out.printf("Hardware address: %s\n",
                Arrays.toString(netint.getHardwareAddress()));
        out.printf("MTU: %s\n", netint.getMTU());
        out.printf("\n");
    }
}
```

(Bentley, TutorialAtOracle Code Examples, 2013)
C:\Users\John\Documents\Sda\Code\Java\Examples\TutorialAtOracle\HelloWorld\src\helloworld\net\NetworkInterfaceDemo.java
(Oracle, 2017) https://docs.oracle.com/javase/tutorial/networking/nifs/parameters.html

View Network Interfaces/adapters in windows with:

- Windows Search > " System Information" > System Summary > Components > Network > Adapter; or
- {Speccy} > Network

Cookies

Cookies enable the saving of state: for a web session, or across web sessions. By themselves HTTP requests and responses can't save state.

(Oracle, 2017) https://docs.oracle.com/javase/tutorial/networking/cookies/definition.html

The standard for managing state with HTTP, and so cookies, is defined at http://www.ietf.org/rfc/rfc2965.txt

Default cookies with the default cookie manager. Cookies will only persist for the session.

```
static private void CookieOperationsDefault () throws URISyntaxException {
    CookieManager cm = new CookieManager();
    CookieManager.setDefault(cm);
    CookieStore store = cm.getCookieStore();

    URI uri = new URI("http://www.examplejohn.com");

    HttpCookie cookie01 = new HttpCookie("jlbvar01", "crazytimes");
    store.add(uri,cookie01);

    HttpCookie cookie02 = new HttpCookie("jlbvar02", "funtimes");
    store.add(uri,cookie02);

    List<HttpCookie> cookies = store.getCookies();
    for (HttpCookie cookie: cookies){
```

```
        System.out.printf(
                "%s: %S (%s)%n", cookie.getName(), cookie.getValue(), cookie.getPath());
    }

    System.out.println("\nURIs:");
    List<URI> uris = store.getURIs();
    for (URI uriloop : uris) {
        System.out.println(uriloop);
    }
    System.out.println();

    System.out.println("Retrieve cookie using custom getCookieValue():");
    System.out.printf("  %s: %s\n", "jlbvar01", getCookieValue(store, "jlbvar01"));
    System.out.printf("  %s: %s\n", "jlbvar02", getCookieValue(store, "jlbvar02"));
}

static public String getCookieValue (CookieStore store, String cookieName){
    List<HttpCookie> cookies = store.getCookies();
    for (HttpCookie cookie: cookies){
        if (cookie.getName().equalsIgnoreCase(cookieName)) {
            return cookie.getValue();
        }
    }
    return "[not found]";
}
```

(Bentley, EcmaScript in Web Quick Reference, 2004)
C:\Users\John\Documents\Sda\Code\Java\Examples\TutorialAtOracle\HelloWorld\src\helloworld\net\cookies\CookieDemo.java

To get cookies to persist between sessions (by writing to disk) implement the `CookieStore` class. See ...

(Oracle, 2017) , "Custom CookieManager", https://docs.oracle.com/javase/tutorial/networking/cookies/custom.html

To enforce a custom CookiePolicy implement the `CookiePolicy` class. see ...

(Oracle, 2017) , "Custom CookieManager", https://docs.oracle.com/javase/tutorial/networking/cookies/custom.html

Reflection

Intro

Every type is a reference or a primitive.

Reference types include:

- Classes (e.g. java.lang.String), enums (e.g. javax.swing.SortOrder), interfaces (e.g. java.io.Serializable), and arrays (all inheriting from java.lang.Object).
- Class wrappers for primitive types (e.g. java.lang.Double).

The primitive types are: `boolean, byte, short, int, long, char, float,` and `double.`

(Oracle, 2017), " Lesson: Classes", https://docs.oracle.com/javase/tutorial/reflect/class/index.html

Get class instance

For every type there exists `java.lang.Class` which provides the entry point for reflection.

(Oracle, 2017), " Lesson: Classes", https://docs.oracle.com/javase/tutorial/reflect/class/index.html

There are several ways to get a `java.lang.Class` instance:

(Oracle, 2017), "Retrieving Class Objects", https://docs.oracle.com/javase/tutorial/reflect/class/classNew.html ...

- If an instance of a reference type is available use the `getClass()` method:
  - From a class instance ...

```
String myString = "Cool";
Class c = myString.getClass();
System.out.println(c.getCanonicalName());

// java.lang.String
```

  - From an enum constant ...

```
public enum Gender { MALE, FEMALE}
static private void getClassObjectFromEnumConstant() {
    // From enum constant
    Class c = Gender.MALE.getClass();
    System.out.println(c.getCanonicalName());
}

// helloworld.demo.ReflectionDemo.Gender
```

  - From an array instance ...

```
int[] scores = { 10, 5, 3, 6};
System.out.println(Arrays.toString(scores));
Class c = scores.getClass();
System.out.println(c.getCanonicalName());

// [10, 5, 3, 6]
// int[]
```

- From an implementation to a collection interface

```
java.util.Set<String> s = new java.util.HashSet<String>();
Class c = s.getClass();
System.out.println(c.getCanonicalName());

// java.util.HashSet
```

- Get class directly from a type, reference or primitive, using `.class` (Dot Class) syntax.

```
// Reference type
Class c = String.class;
System.out.println(c.getCanonicalName());
// java.lang.String

// Primative type
c = int.class;
System.out.println(c.getCanonicalName());
// int

// Multidimentional array of a type
c = int[][].class;
System.out.println(c.getCanonicalName());
// int[][]
```

- If a fully qualified class name is available, you can use `Class.forName()`. Not available for primitive types.

  The string syntax is supposed to be defined in the `Class.getName()` method (Oracle, 2017)
  https://docs.oracle.com/javase/8/docs/api/java/lang/Class.html#getName--

> If this class object represents a class of arrays, then the internal form of the name consists of the name of the element type preceded by one or more '[' characters representing the depth of the array nesting. The encoding of element type names is as follows:
>
> | Element Type | Encoding |
> |---|---|
> | boolean | Z |
> | byte | B |
> | char | C |
> | class or interface | Lclassname; |
> | double | D |
> | float | F |
> | int | I |
> | long | J |
> | short | S |
>
> ...
>
> Examples:
>
> String.class.getName()
> returns "java.lang.String"
> byte.class.getName()
> returns "byte"
> (new Object[3]).getClass().getName()
> returns "[Ljava.lang.Object;"
> (new int[3][4][5][6][7][8][9]).getClass().getName()
> returns "[[[[[[[I"

```
Class c = Class.forName("java.net.Inet6Address");
System.out.println(c.getName());
// java.net.Inet6Address

Class cDoubleArray = Class.forName("[D");
System.out.println(cDoubleArray.getName());
// [D

Class cStringMultidimensionalArray = Class.forName("[[Ljava.lang.String");
System.out.println(cStringMultidimensionalArray.getName());
// Should return [[Ljava/lang/String but returns a java.lang.ClassNotFoundException
```

- For primitive type wrappers, and void, you can use the `TYPE` field. Although for primitives it is preferred that you use the .Class syntax.

```
Class c = Double.TYPE;
System.out.println(c.getName());
// double

c = Void.TYPE;
System.out.println(c.getName());
// void
```

- Methods that return classes ...
    - Get superclass

```
Class c = javax.swing.JButton.class.getSuperclass();
System.out.println(c.getName());
// javax.swing.AbstractButton
```

    - From a class get all public member classes, interfaces, and enums (includes inherited members)

```
Class<?>[] publicClasses = Character.class.getClasses();
for (Class<?> cLoop : publicClasses) {
    System.out.println(cLoop.getName());
}
//  java.lang.Character$Subset
//  java.lang.Character$UnicodeBlock
//  java.lang.Character$UnicodeScript
```

    - From a class get all (public or private) member classes, interfaces, and enums - explicitly declared in this class

```
Class<?>[] allClasses = Character.class.getDeclaredClasses();
for (Class<?> cLoop : allClasses) {
    System.out.println(cLoop.getName());
}
//    java.lang.Character$CharacterCache
//    java.lang.Character$Subset
//    java.lang.Character$UnicodeBlock
//    java.lang.Character$UnicodeScript
```

    - Get the class in which an instance member is declared.

```
//    Class.getDeclaringClass()
//    java.lang.reflect.Field.getEnclosingClass()()
//    java.lang.reflect.Method.getDeclaringClass()
//    java.lang.reflect.Constructor.getDeclaringClass()
Field f = System.class.getField("out");
c = f.getDeclaringClass();
System.out.println(c.getName());
// java.lang.System
```

- Get the class in which an anonymous class is defined. We must use getEnclosingClass() not getEnclosingClass().

```
package helloworld.demo.reflection;

public class AnonymousClassDemo {
    static Object o = new Object() {
        public void m() {}
    };
    static Class<?> c = o.getClass().getEnclosingClass();
}

System.out.println(AnonymousClassDemo.c.getName());
// helloworld.demo.reflection.AnonymousClassDemo
```

- Get the "immediately enclosing class" of a class (in this case an enum class) ...

```
c = Thread.State.class.getEnclosingClass();
System.out.println(c.getName());
// java.lang.Thread
```

(Oracle, 2017) , "Retrieving Class Objects", https://docs.oracle.com/javase/tutorial/reflect/class/classNew.html
C:\Users\John\Documents\Sda\Code\Java\Examples\TutorialAtOracle\HelloWorld\src\helloworld\demo\reflection\GetClass.java
C:\Users\John\Documents\Sda\Code\Java\Examples\TutorialAtOracle\HelloWorld\src\helloworld\demo\reflection\AnonymousClassDemo.java

Get class modifiers, parameter types, and annotations

See (Bentley, Java Reference - Language.docx, 2017) , "Class Modifiers".

Use `java.lang.reflect.Modifier`; and `Class.getModifiers()`.

(Oracle, 2017) , "Examining Class Modifiers and Types", https://docs.oracle.com/javase/tutorial/reflect/class/classModifiers.html

To get class modifiers, parameter types, and annotations ...

```
public class GetClassModifiersParameterTypesAnnotations {
    static public void start()  {
        // A dummy declaration just to look up structure with ctrl + H, ctrl + B
        java.util.concurrent.ConcurrentNavigableMap cvn;

        ClassDeclarationSpy("java.util.concurrent.ConcurrentNavigableMap");
        ClassDeclarationSpy("[Ljava.lang.String;");
        ClassDeclarationSpy("java.io.InterruptedIOException");
        ClassDeclarationSpy("java.security.Identity");
    }

    private static void printAncestor(Class<?> c, List<Class> l) {
        Class<?> ancestor = c.getSuperclass();
        if (ancestor != null) {
            l.add(ancestor);
            printAncestor(ancestor, l);
        }
    }

    static private void ClassDeclarationSpy(String className) {
        try {
            Class c = Class.forName(className);

            out.format("%n%n*********************************%n");
            out.printf("Class:%n  %s%n%n", c.getCanonicalName());
            out.printf("Modifiers:%n  %s%n%n", Modifier.toString(c.getModifiers()));
```

```
            out.printf("Type Parameters:%n");
            TypeVariable[] tv = c.getTypeParameters();
            if (tv.length != 0) {
                out.format("  ");
                for (TypeVariable t : tv) {
                    out.format("%s ", t.getName());
                }
                out.format("%n%n");
            } else {
                out.format("  -- No Type Parameters --%n%n");
            }

            out.format("Implemented Interfaces:%n");
            Type[] intfs = c.getGenericInterfaces();
            if (intfs.length != 0) {
                for (Type intf : intfs)
                    out.format("  %s%n", intf.toString());
                out.format("%n");
            } else {
                out.format("  -- No Implemented Interfaces --%n%n");
            }

            out.format("Inheritance Path:%n");
            List<Class> l = new ArrayList<Class>();
            printAncestor(c, l);
            if (l.size() != 0) {
                for (Class<?> cl : l)
                    out.format("  %s%n", cl.getCanonicalName());
                out.format("%n");
            } else {
                out.format("  -- No Super Classes --%n%n");
            }

            out.format("Annotations:%n");
            Annotation[] ann = c.getAnnotations();
            if (ann.length != 0) {
                for (Annotation a : ann)
                    out.format("  %s%n", a.toString());
                out.format("%n");
            } else {
                out.format("  -- No Annotations --%n%n");
            }

        } catch (ClassNotFoundException x) {
            x.printStackTrace();
        }
    }
}

// E.g. Output for ClassDeclarationSpy("java.security.Identity");
*******************************
Class:
  java.security.Identity

Modifiers:
  public abstract

Type Parameters:
  -- No Type Parameters --

Implemented Interfaces:
  interface java.security.Principal
  interface java.io.Serializable

Inheritance Path:
  java.lang.Object

Annotations:
  @java.lang.Deprecated()
```

(Oracle, 2017) , "Examining Class Modifiers and Types",
https://docs.oracle.com/javase/tutorial/reflect/class/classModifiers.html
C:\Users\John\Documents\Sda\Code\Java\Examples\TutorialAtOracle\HelloWorld\src\helloworld\demo\reflection\GetClassModifiersParameterTypesAnnotations.java

Get class members

For reflection purposes class members are categorized into: fields, methods, and contructors.

(Oracle, 2017), "Discovering Class Members", https://docs.oracle.com/javase/tutorial/reflect/class/classMembers.html

Reflection methods which return class members either return:

- A search result for a particular member (e.g. `getField()`); or
- A group of members (e.g. `getFields()`).

(Oracle, 2017), "Discovering Class Members", https://docs.oracle.com/javase/tutorial/reflect/class/classMembers.html

Reflection methods range over class members which are one or more of: declared directly in the class; private members; superinterfaces or superclasses for inherited members.

Class Methods for Locating Fields			
Class API	**List of members?**	**Inherited members?**	**Private members?**
getDeclaredField()	no	no	yes
getField()	no	yes	no
getDeclaredFields()	yes	no	yes
getFields()	yes	yes	no

Class Methods for Locating Methods			
Class API	**List of members?**	**Inherited members?**	**Private members?**
getDeclaredMethod()	no	no	yes
getMethod()	no	yes	no
getDeclaredMethods()	yes	no	yes
getMethods()	yes	yes	no

Class Methods for Locating Constructors			
Class API	**List of members?**	**Inherited members?**	**Private members?**
getDeclaredConstructor()	no	N/A1	yes
getConstructor()	no	N/A1	no
getDeclaredConstructors()	yes	N/A1	yes
getConstructors()	yes	N/A1	no

(Oracle, 2017), "Discovering Class Members", https://docs.oracle.com/javase/tutorial/reflect/class/classMembers.html

Get class members code.

```
public class GetClassMembers {

    public enum ClassMember {CONSTRUCTOR, FIELD, METHOD, CLASS, ALL}

    public static void start() {
//        OutputClassMembers("java.nio.channels.ReadableByteChannel");
//        java.nio.channels.ReadableByteChannel rbc;
        OutputClassMembers("helloworld.demo.reflection.GetClassMembers");
    }

    private static void printMembers(Member[] members, String heading) {
        out.format("%s:%n", heading);

        for (Member member : members) {
            if (member instanceof Field) {
                out.format("  %s%n", ((Field) member).toGenericString());
            } else if (member instanceof Constructor) {
                out.format("  %s%n", ((Constructor) member).toGenericString());
            } else if (member instanceof Method) {
```

```
                out.format("  %s%n", ((Method) member).toGenericString());
            } else {
                try {
                    throw new Exception("Unexpected else in member if .. else: " +
member.toString());
                } catch (Exception e) {
                    e.printStackTrace();
                }
            }
        } // for
        if (members.length == 0) {
            out.format("  -- No %s --%n", heading);
        }
        out.println();
    }

    private static void printClasses(Class<?> c) {
        out.printf("Classes:%n");
        Class<?>[] classes = c.getClasses();
        if (classes.length == 0) {
            out.printf("  -- No member classes, enums, or interfacess --%n");
        } else {
            for (Class<?> cLoop : classes) {
                out.printf("  %s%n", cLoop.getCanonicalName());
            }
        }
        out.println();
    }

    private static void OutputClassMembers(String className) {
        OutputClassMembers(className, ClassMember.ALL);
    }

    private static void OutputClassMembers(String className, ClassMember classMember) {
        try {
            Class<?> c = Class.forName(className);
            out.format("Class:%n  %s%n%n", c.getCanonicalName());

            Package p = c.getPackage();
            out.format("Package:%n  %s%n%n",
                    p != null ? p.getName() : "-- No Package --");

            switch (classMember) {
                case CONSTRUCTOR:
                    printMembers(c.getConstructors(), "Constructor");
                    break;
                case FIELD:
                    printMembers(c.getFields(), "Fields");
                    break;
                case METHOD:
                    printMembers(c.getMethods(), "Methods");
                    break;
                case CLASS:
                    printClasses(c);
                    break;
                case ALL:
                    printMembers(c.getConstructors(), "Constructor");
                    printMembers(c.getFields(), "Fields");
                    printMembers(c.getMethods(), "Methods");
                    printClasses(c);
                    break;
                default:
                    assert false;
            }

        } catch (ClassNotFoundException e) {
            e.printStackTrace();
        }
    }
}

// Output
Class:
  helloworld.demo.reflection.GetClassMembers

Package:
  helloworld.demo.reflection
```

```
Constructor:
  public helloworld.demo.reflection.GetClassMembers()

Fields:
  -- No Fields --

Methods:
  public static void helloworld.demo.reflection.GetClassMembers.start()
  public final void java.lang.Object.wait() throws java.lang.InterruptedException
  public final void java.lang.Object.wait(long,int) throws java.lang.InterruptedException
  public final native void java.lang.Object.wait(long) throws java.lang.InterruptedException
  public boolean java.lang.Object.equals(java.lang.Object)
  public java.lang.String java.lang.Object.toString()
  public native int java.lang.Object.hashCode()
  public final native java.lang.Class<?> java.lang.Object.getClass()
  public final native void java.lang.Object.notify()
  public final native void java.lang.Object.notifyAll()

Classes:
  helloworld.demo.reflection.GetClassMembers.ClassMember
```

(Oracle, 2017), "Discovering Class Members", https://docs.oracle.com/javase/tutorial/reflect/class/classMembers.html
C:\Users\John\Documents\Sda\Code\Java\Examples\TutorialAtOracle\HelloWorld\src\helloworld\demo\reflection\GetClassMembers.java

Get class member details

Fields

For reflection purposes a class field is either:

- A class;
- An interface;
- A variable (?);
- An enum with associated value.

(Oracle, 2017), Fields", https://docs.oracle.com/javase/tutorial/reflect/member/field.html

A field may be an instance or static field.

(Oracle, 2017), Fields", https://docs.oracle.com/javase/tutorial/reflect/member/field.html

Get the field type with `getType()` or `getGenericType()`.

```
public class GetFieldDetails<T> {

    public boolean[][] b = {{ false, false }, { true, true } };
    public String name  = "Alice";
    public List<Integer> list;
    public T val;

    public static void start() {
         OutputFieldType("helloworld.demo.reflection.GetFieldDetails", "list");
    }

    private static void OutputFieldType(String className, String fieldName){
        Class<?> c = null;
        try {
            c = Class.forName(className);
            Field f = c.getField(fieldName);

            out.printf("Class: %s%n", c.getCanonicalName());
            out.printf("  Field: %s%n", f.getName());
            out.printf("    Type: %s%n", f.getType());
```

```
            out.printf("    Generic Type: %s%n", f.getGenericType());
            out.println();

        } catch (ClassNotFoundException e) {
            e.printStackTrace();
        } catch (NoSuchFieldException e) {
            e.printStackTrace();
        }
    }
}

// Output
Class: helloworld.demo.reflection.GetFieldDetails
  Field: list
    Type: interface java.util.List
    Generic Type: java.util.List<java.lang.Integer>
```

(Oracle, 2017), "Obtaining Field Types", https://docs.oracle.com/javase/tutorial/reflect/member/fieldTypes.html
C:\Users\John\Documents\Sda\Code\Java\Examples\TutorialAtOracle\HelloWorld\src\helloworld\demo\reflection\GetFieldDetails.java

Field modifier kinds:

- Access: `public`, `protected` and `private`.
- Runtime behaviour: `transient` and `volatile`.
- One instance: `static`.
- Make a constant: `final`.
- Annotation.

(Oracle, 2017), " Retrieving and Parsing Field Modifiers", https://docs.oracle.com/javase/tutorial/reflect/member/fieldModifiers.html

Return fields with specified modifiers.

```
public class GetFieldDetails<T> {

    public enum Spy { BLACK , WHITE }
    public boolean[][] b = {{ false, false }, { true, true } };
    public String name  = "Alice";
    public List<Integer> list;
    public T val;
    public volatile int share;
    int instance;
    class Inner {}

    public static void start() {

        String className = "helloworld.demo.reflection.GetFieldDetails";

        OutputFieldModifiers(className, "volatile", "public");
        OutputFieldModifiers(className, "public");
        OutputFieldModifiers(className, "protected");
    }

    private static int modifierFromString(String s) {
        int m = 0x0;
        switch(s){
            case "public":
                m |= Modifier.PUBLIC;
                break;
            case "protected":
                m |= Modifier.PROTECTED;
                break;
            case "private":
                m |= Modifier.PRIVATE;
                break;
            case "static":
                m |= Modifier.STATIC;
                break;
```

```
            case "final":
                m |= Modifier.FINAL;
                break;
            case "transient":
                m |= Modifier.TRANSIENT;
                break;
            case "volatile":
                m |= Modifier.VOLATILE;
                break;
            default:
                try {
                    throw new Exception("Unexpected case in modifierFromString. s = " + s);
                } catch (Exception e) {
                    e.printStackTrace();
                }
        }
        return m;
    }

    private static void OutputFieldModifiers(String className, String ... hasAllModifiers){

        try {
            Class<?> c = Class.forName(className);
            int searchModifiers = 0x0;

            for (int i = 0; i < hasAllModifiers.length; i++) {
                searchModifiers |= modifierFromString(hasAllModifiers[i]);
            }

            Field[] fields = c.getDeclaredFields();
            out.printf("Fields in Class '%s' having all of the modifiers: %s%n",
                    c.getName(),
                    Modifier.toString(searchModifiers));

            boolean found = false;
            for (Field f : fields) {
                int foundModifiers = f.getModifiers();
                if ((foundModifiers & searchModifiers) == searchModifiers) {
                    out.printf("%-8s [ synthetic=%-5b enum_constant=%-5b ]%n",
                            f.getName(),
                            f.isSynthetic(),
                            f.isEnumConstant());
                    found = true;
                }
            }

            if (!found) {
                out.printf("No matching fields");
            }
            out.println();

        } catch (ClassNotFoundException e) {
            e.printStackTrace();
        }
    }
}

// Output
Fields in Class 'helloworld.demo.reflection.GetFieldDetails' having all of the modifiers:
public volatile
share    [ synthetic=false enum_constant=false ]

Fields in Class 'helloworld.demo.reflection.GetFieldDetails' having all of the modifiers:
public
b        [ synthetic=false enum_constant=false ]
name     [ synthetic=false enum_constant=false ]
list     [ synthetic=false enum_constant=false ]
val      [ synthetic=false enum_constant=false ]
share    [ synthetic=false enum_constant=false ]

Fields in Class 'helloworld.demo.reflection.GetFieldDetails' having all of the modifiers:
protected
No matching fields
```

C:\Users\John\Documents\Sda\Code\Java\Examples\TutorialAtOracle\HelloWorld\src\helloworld\demo\reflection\GetFieldDetails.java

(Oracle, 2017) ," Retrieving and Parsing Field Modifiers",
https://docs.oracle.com/javase/tutorial/reflect/member/fieldModifiers.html

Get and set field values.

```
// Given
enum Tweedle { DEE, DUM };

public class Book {
    public long chapters = 0;
    public String[] characters = { "Alice", "White Rabbit" };
    public Tweedle twin = Tweedle.DEE;
}

// Work code
/ Dependency: Book class
public class GetFieldValuesAndSet {

    public static void start() {
        DemoGetFieldValuesAndSet();
    }

    public static void DemoGetFieldValuesAndSet() {
        Book book = new Book();
        Class<?> c = book.getClass();

        String format = "%6S:  %-12s = %s%n";

        try {
            // A long
            Field chaptersReflect = c.getDeclaredField("chapters");
            out.printf(format, "before", "chapters", book.chapters);
            chaptersReflect.setLong(book, 12);
            out.printf(format, "after", "chapters", chaptersReflect.getLong(book));

            // An Array
            Field charactersReflect = c.getDeclaredField("characters");
            out.printf(format, "before", "characters[]", Arrays.asList(book.characters));
            String[] newCharacters = {"Queen", "King"};
            charactersReflect.set(book, newCharacters);
            out.printf(format, "after", "characters[]", Arrays.asList(book.characters));

            // Enum
            Field twinReflect = c.getDeclaredField("twin");
            out.printf(format, "before", "twin", book.twin);
            twinReflect.set(book, Tweedle.DUM);
            out.printf(format, "after", "twin", twinReflect.get(book));

        } catch (NoSuchFieldException e) {
            e.printStackTrace();
        } catch (IllegalAccessException e) {
            e.printStackTrace();
        }

    }
}

// Output
BEFORE:  chapters     = 0
 AFTER:  chapters     = 12
BEFORE:  characters[] = [Alice, White Rabbit]
 AFTER:  characters[] = [Queen, King]
BEFORE:  twin         = DEE
 AFTER:  twin         = DUM
```

C:\Users\John\Documents\Sda\Code\Java\Examples\TutorialAtOracle\HelloWorld\src\helloworld\demo\reflection\GetFieldValuesAndSet.java
(Oracle, 2017), "Getting and Setting Field Values",
https://docs.oracle.com/javase/tutorial/reflect/member/fieldValues.html

Methods

Return types, parameter types, exception types

Get method: return types; parameter types; and exception types.

```
public class GetMethodDetails {

    public static void start() {
        OutputMethodType("java.lang.String", "offsetByCodePoints");
        OutputMethodType("java.lang.Class", "getConstructor");
        OutputMethodType("java.io.PrintStream", "format");
    }

    private static void OutputMethodType(String className, String methodName){
        final String format = "%24s: %s%n";

        out.println();
        try {
            Class<?> c = Class.forName(className);
            Method[] allMethods = c.getDeclaredMethods();
            for (Method m : allMethods) {
                if (!m.getName().equals(methodName)) {
                    continue;
                }
                out.println(m.toGenericString());
                out.format(format, "ReturnType", m.getReturnType());
                out.format(format, "GenericReturnType", m.getGenericReturnType());

                Class<?>[] parameterTypes = m.getParameterTypes();
                Type[] genericParameterTypes = m.getGenericParameterTypes();
                 for (int i = 0; i < parameterTypes.length; i++) {
                    out.format(format, "ParameterType", parameterTypes[i]);
                    out.format(format, "GenericParameterType", genericParameterTypes[i]);
                }

                Class<?>[] exceptionTypes = m.getExceptionTypes();
                Type[] genericExceptionTypes = m.getGenericExceptionTypes();
                for (int i = 0; i < exceptionTypes.length; i++) {
                    out.format(format, "ExceptionType", exceptionTypes[i]);
                    out.format(format, "GenericExceptionType", genericExceptionTypes[i]);
                }
             }
        } catch (ClassNotFoundException e) {
            e.printStackTrace();
        }
    }
}

// Output
public int java.lang.String.offsetByCodePoints(int,int)
              ReturnType: int
       GenericReturnType: int
           ParameterType: int
    GenericParameterType: int
           ParameterType: int
    GenericParameterType: int

public java.lang.reflect.Constructor<T>
java.lang.Class.getConstructor(java.lang.Class<?>...) throws
java.lang.NoSuchMethodException,java.lang.SecurityException
              ReturnType: class java.lang.reflect.Constructor
       GenericReturnType: java.lang.reflect.Constructor<T>
           ParameterType: class [Ljava.lang.Class;
    GenericParameterType: java.lang.Class<?>[]
           ExceptionType: class java.lang.NoSuchMethodException
    GenericExceptionType: class java.lang.NoSuchMethodException
           ExceptionType: class java.lang.SecurityException
    GenericExceptionType: class java.lang.SecurityException

public java.io.PrintStream
java.io.PrintStream.format(java.util.Locale,java.lang.String,java.lang.Object...)
              ReturnType: class java.io.PrintStream
```

```
       GenericReturnType: class java.io.PrintStream
           ParameterType: class java.util.Locale
    GenericParameterType: class java.util.Locale
           ParameterType: class java.lang.String
    GenericParameterType: class java.lang.String
           ParameterType: class [Ljava.lang.Object;
    GenericParameterType: class [Ljava.lang.Object;
public java.io.PrintStream java.io.PrintStream.format(java.lang.String,java.lang.Object...)
              ReturnType: class java.io.PrintStream
       GenericReturnType: class java.io.PrintStream
           ParameterType: class java.lang.String
    GenericParameterType: class java.lang.String
           ParameterType: class [Ljava.lang.Object;
    GenericParameterType: class [Ljava.lang.Object;
```

C:\Users\John\Documents\Sda\Code\Java\Examples\TutorialAtOracle\HelloWorld\src\helloworld\demo\reflection\GetMethodDetails.java
(Oracle, 2017) , "Obtaining Method Type Information",
https://docs.oracle.com/javase/tutorial/reflect/member/methodType.html

Parameter names

To obtain the method names of parameters, you need code like this ...

```
public class GetMethodDetails {

    private static final String format = "%24s: %s%n";

    public static void start() {
        outputClassMethods("helloworld.demo.reflection.examples.ExampleMethods");
    }

    private static void outputParameter(Parameter p) {
        out.format(format, "Parameter name", p.getName());
        out.format(format, "Parameter type", p.getType());
        out.format(format, "Modifiers", p.getModifiers());
        out.format(format, "Is implicit?", p.isImplicit());
        out.format(format, "Is name present?", p.isNamePresent());
        out.format(format, "Is synthetic", p.isSynthetic());
    }

    private static void outputMethod(Method m) {
        out.format("%n%s%n", m.toGenericString());
        out.format(format, "Return Type", m.getReturnType());
        out.format(format, "Generic Return Type", m.getGenericReturnType());

        Parameter[] parameters = m.getParameters();
        for (Parameter p : parameters) {
            outputParameter(p);
        }
    }

    private static void outputClassMethods(String className) {
        Class<?> c = null;
        try {
            c = Class.forName(className);
        } catch (ClassNotFoundException e) {
            e.printStackTrace();
        }
        Method[] methods = c.getDeclaredMethods();
        out.format(format, "Number of methods", methods.length);
        for (Method m : methods) {
            outputMethod(m);
        }
    }
}

// Given
package helloworld.demo.reflection;

import java.util.*;
```

```
public class ExampleMethods<T> {

    public boolean simpleMethod(String stringParam, int intParam) {
        System.out.println("String: " + stringParam + ", integer: " + intParam);
        return true;
    }

    public int varArgsMethod(String... manyStrings) {
        return manyStrings.length;
    }

    public boolean methodWithList(List<String> listParam) {
        return listParam.isEmpty();
    }

    public <T> void genericMethod(T[] a, Collection<T> c) {
        System.out.println("Length of array: " + a.length);
        System.out.println("Size of collection: " + c.size());
    }

}

// Output
     Number of methods: 4

public boolean
helloworld.demo.reflection.examples.ExampleMethods.methodWithList(java.util.List<java.lang.S
tring>)
             Return Type: boolean
     Generic Return Type: boolean
          Parameter name: listParam
          Parameter type: interface java.util.List
               Modifiers: 0
            Is implicit?: false
        Is name present?: true
            Is synthetic: false

public int
helloworld.demo.reflection.examples.ExampleMethods.varArgsMethod(java.lang.String...)
             Return Type: int
     Generic Return Type: int
          Parameter name: manyStrings
          Parameter type: class [Ljava.lang.String;
               Modifiers: 0
            Is implicit?: false
        Is name present?: true
            Is synthetic: false

public boolean
helloworld.demo.reflection.examples.ExampleMethods.simpleMethod(java.lang.String,int)
             Return Type: boolean
     Generic Return Type: boolean
          Parameter name: stringParam
          Parameter type: class java.lang.String
               Modifiers: 0
            Is implicit?: false
        Is name present?: true
            Is synthetic: false
          Parameter name: intParam
          Parameter type: int
               Modifiers: 0
            Is implicit?: false
        Is name present?: true
            Is synthetic: false

public <T> void
helloworld.demo.reflection.examples.ExampleMethods.genericMethod(T[],java.util.Collection<T>
)
             Return Type: void
     Generic Return Type: void
          Parameter name: a
          Parameter type: class [Ljava.lang.Object;
               Modifiers: 0
            Is implicit?: false
        Is name present?: true
            Is synthetic: false
          Parameter name: c
```

```
          Parameter type: interface java.util.Collection
              Modifiers: 0
            Is implicit?: false
        Is name present?: true
            Is synthetic: false
```

(Oracle, 2017), " Obtaining Names of Method Parameters",
https://docs.oracle.com/javase/tutorial/reflect/member/methodparameterreflection.html
C:\Users\John\Documents\Sda\Code\Java\Examples\TutorialAtOracle\HelloWorld\src\helloworld\demo\reflection\GetMethodDetails.java

... and you need to pass the `-parameters` option to the `javac` compiler.

- To do this from within IntelliJ IDEA:
  - File > Settings ... > Build, Execution, Deployment > Java Compiler > -Javac Options- > Override compiler parameters per-module.
  - [+] > Module: MyProject > Compilation options: "-parameters".

There are security and performance reasons for not exposing method parameter names to the reflection API by default.

(Oracle, 2017), " Obtaining Names of Method Parameters",
https://docs.oracle.com/javase/tutorial/reflect/member/methodparameterreflection.html

Implicit and Synthetic Constructs

Implicit constructs. Class constructors and parameters can be implicitly declared by the java compiler.

Synethetic constructs. A synthetic construct is one which is not explicitly declared nor "implicitly [declared] implicitly in source code, unless they are class initialization methods".

The formal parameters for the default constructor of an enum construct are not implicitly declared because different compilers need not agree on the form of this constructor; another Java compiler might specify different formal parameters for it. When compilers compile expressions that use enum constants, they rely only on the public static fields of the enum construct, which are implicitly declared, and not on their constructors or how these constants are initialized.

(Oracle, 2017), " Obtaining Names of Method Parameters",
https://docs.oracle.com/javase/tutorial/reflect/member/methodparameterreflection.html

Method modifiers

There a several modifiers that may be part of a method declaration:

- Access modifiers: public, protected, and private
- Modifier restricting to one instance: static
- Modifier prohibiting value modification: final
- Modifier requiring override: abstract
- Modifier preventing reentrancy: synchronized
- Modifier indicating implementation in another programming language: native
- Modifier forcing strict floating point behavior: strictfp
- Annotations

(Oracle, 2017), " Obtaining Names of Method Parameters",
https://docs.oracle.com/javase/tutorial/reflect/member/methodparameterreflection.html

Get method modifiers and whether the method is: synthetic, has variable args, a bridge method.

```
private static void outputMethodModifiers(Method m) {
    out.format(format, "Modifiers section ..... ", "");
    out.format(format, "Modifiers", Modifier.toString(m.getModifiers()));
    out.format("  [ synthetic=%-5b var_args=%-5b bridge=%-5b ]%n",
            m.isSynthetic(),
            m.isVarArgs(),
            m.isBridge());
}

// Output
public int
helloworld.demo.reflection.examples.ExampleMethods.varArgsMethod(java.lang.String...)
...
Modifiers section ..... :
               Modifiers: public transient
  [ synthetic=false var_args=true  bridge=false ]
```

(Oracle, 2017), " Obtaining Names of Method Parameters",
https://docs.oracle.com/javase/tutorial/reflect/member/methodparameterreflection.html
C:\Users\John\Documents\Sda\Code\Java\Examples\TutorialAtOracle\HelloWorld\src\helloworld\demo\reflection\GetMethodDetails.java

Invoking

Invoke a method with a fixed number of arguments by settting it to be accessabile and using the invoke method, passing: a class instance and the method paramaters.

```
m.setAccessible(true);
Object obj = m.invoke(classInstance, new Locale(language, country, variant));

// As in
private static void invokeMethodWithFixedArgs() {

    String language = "ja";
    String country = "JP";
    String variant = "JP";

    try {
        Class<?> c = Class.forName("helloworld.demo.reflection.examples.Deet");
        Object classInstance = c.getDeclaredConstructor().newInstance();

        Method[] declaredMethods = c.getDeclaredMethods();
        for (Method m : declaredMethods) {
            String methodName = m.getName();
            if (!methodName.startsWith("test")
                    || (m.getGenericReturnType() != boolean.class)) {
                continue;
            }

            Type[] parameterTypes = m.getGenericParameterTypes();
            if ((parameterTypes.length != 1)
                    || Locale.class.isAssignableFrom(parameterTypes[0].getClass())) {
                continue;
            }

            out.format("Invoking %s()%n", methodName);
            try {
                m.setAccessible(true);
                Object obj = m.invoke(classInstance, new Locale(language, country,
variant));
                out.format("%s() returned %b%n", methodName, (Boolean) obj);
            } catch (InvocationTargetException e) {
                Throwable cause = e.getCause();
                err.format("invocation of %s failed: %s%n",
                        methodName, cause.getMessage());
            }
```

```
        }

    } catch (ClassNotFoundException e) {
        e.printStackTrace();
    } catch (IllegalAccessException e) {
        e.printStackTrace();
    } catch (InstantiationException e) {
        e.printStackTrace();
    } catch (NoSuchMethodException e) {
        e.printStackTrace();
    } catch (InvocationTargetException e) {
        e.printStackTrace();
    }
}

// Given
public class Deet<T> {
    private boolean testDeet(Locale l) {
        // getISO3Language() may throw a MissingResourceException
        System.out.format("Locale: %s, ISO Language Code: %s%n", l.getDisplayName(),
l.getISO3Language());
        return true;
    }
}

// Outputs
Invoking testDeet()
Locale: Japanese (Japan,JP), ISO Language Code: jpn
testDeet() returned true
```

C:\Users\John\Documents\Sda\Code\Java\Examples\TutorialAtOracle\HelloWorld\src\helloworld\demo\reflection\MethodInvocation.java
C:\Users\John\Documents\Sda\Code\Java\Examples\TutorialAtOracle\HelloWorld\src\helloworld\demo\reflection\examples\Deet.java
(Oracle, 2017), "Invoking Methods", https://docs.oracle.com/javase/tutorial/reflect/member/methodInvocation.html

Invoking a method with a variable number of arugments.

```
private static void invokeMethodWithVarArgs() {

    try {
        Class<?> c = Class.forName("helloworld.demo.reflection.examples.Deet");
        Object classInstance = c.getDeclaredConstructor().newInstance();

        // An array of parameterTypes, containing one item:
        // A parameter of type String array.
        Class[] parameterTypes = new Class[] {String[].class};

        Method m = c.getDeclaredMethod("outputFruit", parameterTypes);
        String methodName = m.getName();
        String[] fruit = {"Apples", "Oranges", "Pears"};

        out.format("Invoking %s()%n", methodName);

        m.setAccessible(true);
        Object obj = m.invoke(classInstance, (Object)fruit);
        out.format("%s() returned %b%n", methodName, (Boolean) obj);

    } catch (ClassNotFoundException
            | IllegalAccessException
            | InstantiationException
            | NoSuchMethodException
            | InvocationTargetException
            e) {
        e.printStackTrace();
    }
}

// Given

public class Deet {
 ...

    public boolean outputFruit(String... args) {
```

```
        for (String s : args) {
            System.out.println(s.toString());
        }
        return true;
    }
}

// Output
Invoking outputFruit()
Apples
Oranges
Pears
outputFruit() returned true
```

C:\Users\John\Documents\Sda\Code\Java\Examples\TutorialAtOracle\HelloWorld\src\helloworld\demo\reflection\MethodInvocation.java

Constructors

Get class constructors.

```
public class GetMethodDetails {

    private static final String format = "%24s: %s%n";

    public static void start() {
        outputClassConstructors("helloworld.demo.reflection.examples.Book");
    }

    private static void outputParameter(Parameter p) {
        out.format(format, "Parameter class", p.getType());
        out.format(format, "Parameter name", p.getName());
        out.format(format, "Modifiers", p.getModifiers());
        out.format(format, "Is implicit?", p.isImplicit());
        out.format(format, "Is name present?", p.isNamePresent());
        out.format(format, "Is synthetic", p.isSynthetic());
    }

    private static void outputConstructor(Constructor constructor) {
        out.format(format,"Constructor", constructor.toGenericString());
        Parameter[] parameters = constructor.getParameters();
        out.format(format, "Number of parameters", parameters.length);
        for (Parameter p : parameters) {
            outputParameter(p);
        }
    }

    private static void outputClassConstructors(String className){
        Class<?> c = null;
        try {
            c = Class.forName(className);
        } catch (ClassNotFoundException e) {
            e.printStackTrace();
        }

        Constructor[] allDeclaredConstructors = c.getDeclaredConstructors();
        out.println();
        out.format(format, "Number of declared constructors",
allDeclaredConstructors.length);
        for (Constructor constructor : allDeclaredConstructors) {
            outputConstructor(constructor);
        }
    }
}

// Given
package helloworld.demo.reflection.examples;

public class Book {
    ...

    public Book(long chapters) {
        mChapters = chapters;
```

```
    }
}

// Output
Number of declared constructors: 1
             Constructor: public helloworld.demo.reflection.examples.Book(long)
    Number of parameters: 1
          Parameter name: chapters
          Parameter type: long
               Modifiers: 0
             Is implicit?: false
        Is name present?: true
             Is synthetic: false
```

C:\Users\John\Documents\Sda\Code\Java\Examples\TutorialAtOracle\HelloWorld\src\helloworld\demo\reflection\GetMethodDetails.java
(Oracle, 2017), "Obtaining Names of Method Parameters",
https://docs.oracle.com/javase/tutorial/reflect/member/methodparameterreflection.html
See also (Oracle, 2017) , Finding Constructors, https://docs.oracle.com/javase/tutorial/reflect/member/ctorLocation.html

Enums are implemented internally as classes.

```
outputMethodsAndConstructors("helloworld.demo.reflection.examples.Tweedle");

private static void outputMethodsAndConstructors(String className) {
    outputClassMethods(className);
    outputClassConstructors(className);
}

// Output
       Number of methods: 2

public static helloworld.demo.reflection.examples.Tweedle[]
helloworld.demo.reflection.examples.Tweedle.values()
             Return Type: class [Lhelloworld.demo.reflection.examples.Tweedle;
     Generic Return Type: class [Lhelloworld.demo.reflection.examples.Tweedle;

public static helloworld.demo.reflection.examples.Tweedle
helloworld.demo.reflection.examples.Tweedle.valueOf(java.lang.String)
             Return Type: class helloworld.demo.reflection.examples.Tweedle
     Generic Return Type: class helloworld.demo.reflection.examples.Tweedle
          Parameter name: name
          Parameter type: class java.lang.String
               Modifiers: 32768
             Is implicit?: true
        Is name present?: true
             Is synthetic: false

Number of declared constructors: 1
             Constructor: private helloworld.demo.reflection.examples.Tweedle()
    Number of parameters: 2
          Parameter name: $enum$name
          Parameter type: class java.lang.String
               Modifiers: 4096
             Is implicit?: false
        Is name present?: true
             Is synthetic: true
          Parameter name: $enum$ordinal
          Parameter type: int
               Modifiers: 4096
             Is implicit?: false
        Is name present?: true
             Is synthetic: true
```

C:\Users\John\Documents\Sda\Code\Java\Examples\TutorialAtOracle\HelloWorld\src\helloworld\demo\reflection\GetMethodDetails.java
(Oracle, 2017), " Obtaining Names of Method Parameters",
https://docs.oracle.com/javase/tutorial/reflect/member/methodparameterreflection.html

Create new class from a constructor, where the constructor has no arguments. Use: `java.lang.reflect.Constructor.newInstance()` rather than `Class.newInstance()`.

```
private static void createInstance() {
    Constructor[] constructors = Console.class.getDeclaredConstructors();
    Constructor constructor = null;
    for (int i = 0; i < constructors.length; i++) {
        constructor = constructors[i];
        if (constructor.getGenericParameterTypes().length == 0) {
            break;
        }

    }

    try {
        constructor.setAccessible(true);
        Console console = (Console) constructor.newInstance();
        Field field = console.getClass().getDeclaredField("cs");
        field.setAccessible(true);

        out.printf("Console charset         : %s%n", field.get(console));
        out.printf("Charset.defaultCharset(): %s%n", Charset.defaultCharset());

    } catch (
            IllegalAccessException e)

    {
        e.printStackTrace();
    } catch (InstantiationException e) {
        e.printStackTrace();
    } catch (InvocationTargetException e) {
        e.printStackTrace();
    } catch (NoSuchFieldException e) {
        e.printStackTrace();
    }
}

// Output
WARNING: An illegal reflective access operation has occurred
WARNING: Illegal reflective access by helloworld.demo.reflection.ConstructorInstanceCreation
(file:/C:/Users/John/Documents/Sda/Code/Java/Examples/TutorialAtOracle/out/production/HelloW
orld/) to constructor java.io.Console()
WARNING: Please consider reporting this to the maintainers of
helloworld.demo.reflection.ConstructorInstanceCreation
WARNING: Use --illegal-access=warn to enable warnings of further illegal reflective access
operations
WARNING: All illegal access operations will be denied in a future release
Console charset         : IBM850
Charset.defaultCharset(): UTF-8
```

C:\Users\John\Documents\Sda\Code\Java\Examples\TutorialAtOracle\HelloWorld\src\helloworld\demo\reflection\ConstructorInstanceCreationNoArgs.java
(Oracle, 2017). "Creating New Class Instances", https://docs.oracle.com/javase/tutorial/reflect/member/ctorInstance.html

In future releases of Java creating instances may not be permitted by default. Then you can work around this by providing a VM option: `--illegal-access=permit`

http://mail.openjdk.java.net/pipermail/jigsaw-dev/2017-May/012673.html
(Stackoverflow, 2018) , JDK9: An illegal reflective access operation has occurred. org.python.core.PySystemState, nullpointer's answer, https://stackoverflow.com/a/46230678/872154

Create new class from a constructor, where the constructor has arguments.

```
// Given a class
public class EmailAliases {
```

```
    private Set<String> aliases;

    private EmailAliases(HashMap<String, String> hashMap) {
        aliases = hashMap.keySet();
    }

    public void printKeys() {
        out.println("Mail keys:");
        for (String key : aliases) {
            out.println("   " + key);
        }
    }
}

// Create an instance with arguments
private static Map<String, String> defaultAliases = new HashMap<String, String>();
static {
    defaultAliases.put("Duke", "duke@i-love-java");
    defaultAliases.put("Fang", "fang@evil-jealous-twin");
}

private static void createInstanceWithArgs(){
    try {
        Constructor constructor = EmailAliases.class.getDeclaredConstructor(HashMap.class);
        constructor.setAccessible(true);
        EmailAliases emailAliases = (EmailAliases) constructor.newInstance(defaultAliases);
        emailAliases.printKeys();

    } catch (NoSuchMethodException e) {
        e.printStackTrace();
    } catch (IllegalAccessException e) {
        e.printStackTrace();
    } catch (InstantiationException e) {
        e.printStackTrace();
    } catch (InvocationTargetException e) {
        e.printStackTrace();
    }
}

// Output
Mail keys:
   Fang
   Duke
```

C:\Users\John\Documents\Sda\Code\Java\Examples\TutorialAtOracle\HelloWorld\src\helloworld\demo\reflection\ConstructorInstanceCreationWithArgs.java
(Oracle, 2017), "Creating New Class Instances", https://docs.oracle.com/javase/tutorial/reflect/member/ctorInstance.html

Arrays and Enumerated Types

Intro

From a reflection point of view arrays and enumerated types (enums) are just classes. Many of the previous techniques can be used on them. However there are some array and enum specific reflection techniques.

(Oracle, 2017), "Lesson: Arrays and Enumerated Types", https://docs.oracle.com/javase/tutorial/reflect/special/index.html

Arrays

Determine whether a field is an array with `Class.isArrary()`.

```
static public void start(){
    outputArrayFieldsInClass("helloworld.demo.reflection.examples.Book");
}

public static void outputArrayFieldsInClass(String className) {
```

```
    boolean found = false;
    try {
        Class<?> cls = Class.forName(className);
        Field[] fields = cls.getDeclaredFields();

        Class<?> fieldClass = null;
        for (Field field : fields) {
            fieldClass = field.getType();
            if (fieldClass.isArray()) {
                found = true;
                System.out.format("%s%n"
                        + "                Field: %s%n"
                        + "                 Type: %s%n"
                        + "        Component Type: %s%n ",
                        field, field.getName(), fieldClass, fieldClass.getComponentType()
                );
            }
        }
        if (!found) {
            System.out.println("No array fields in class");
        }
    } catch (ClassNotFoundException e) {
        e.printStackTrace();
    }
}

// Given
public class Book {
...
    public String[] characters = {"Alice", "White Rabbit"};
    public Tweedle twin = Tweedle.DEE;
...
}

// Output
public java.lang.String[] helloworld.demo.reflection.examples.Book.characters
                Field: characters
                 Type: class [Ljava.lang.String;
        Component Type: class java.lang.String
```

C:\Users\John\Documents\Sda\Code\Java\Examples\TutorialAtOracle\HelloWorld\src\helloworld\demo\reflection\ArrayReflection.java
(Oracle, 2017), "Identifying Array Types", https://docs.oracle.com/javase/tutorial/reflect/special/arrayComponents.html

Create array.

```
private static void createArray() {
    String stringToSearch = "java.math.BigInteger bi[] = { 123, 234, 345 }";
    Pattern pattern = Pattern.compile("^\\s*(\\S+)\\s*\\w+\\[\\].*\\{\\s*([^}]+)\\s*\\}");
    Matcher matcher = pattern.matcher(stringToSearch);

    if (matcher.find()) {
        String className = matcher.group(1);
        String[] classValues = matcher.group(2).split("[\\s,]+");
        int n = classValues.length;

        try{
            Class<?> cls = Class.forName(className);
            Object arrayObject = Array.newInstance(cls, n);
            for (int i = 0; i < n; i++) {
                String value = classValues[i];
                Constructor constructor = cls.getConstructor(String.class);
                Object objectValue = constructor.newInstance(value);
                Array.set(arrayObject, i, objectValue);
            }

            Object[] oo = (Object[])arrayObject;
            System.out.printf("JLB: %s[] = %s%n", className, Arrays.toString(oo));
        } catch (ClassNotFoundException e) {
            e.printStackTrace();
        } catch (NoSuchMethodException e) {
            e.printStackTrace();
        } catch (IllegalAccessException e) {
            e.printStackTrace();
```

```
        } catch (InstantiationException e) {
            e.printStackTrace();
        } catch (InvocationTargetException e) {
            e.printStackTrace();
        }
    }
}
```

C:\Users\John\Documents\Sda\Code\Java\Examples\TutorialAtOracle\HelloWorld\src\helloworld\demo\reflection\ArrayReflection.java
(Oracle, 2017), "Creating New Arrays", https://docs.oracle.com/javase/tutorial/reflect/special/arrayInstance.html

Getting and setting array values can be done as follows:

- To set or get the entire array use:
  - `java.lang.reflect.Field.set(Object obj, Object value);`
  - `java.lang.reflect.Field.get(Object obj);`
- To set or get individual components of the array use:
  - For primitive types, setFoo() or getFoo(), where "foo" is the individual type. These supporting "widening", that is using smaller memoried primivitive types (e.g. Array.getShort()) to set larger memoried primitype types (in this case int).

```
Array.setInt(Object array, int index, int value)
Array.getInt(Object array, int index)
```

  - For reference types, use:
  - `Array.set(Object array, int index, int value)`
  - `Array.get(Object array, int index)`

(Oracle, 2017), "Getting and Setting Arrays and Their Components", https://docs.oracle.com/javase/tutorial/reflect/special/arraySetGet.html

Set Entire Array Eample.

```
private static void setEntireArray(){
    final String ACTION = "grow";
    final int srcBufSize = 10 * 1024;
    char[] src = new char[srcBufSize];
    src[srcBufSize - 1] = 'x';

    CharArrayReader car = new CharArrayReader(src);
    BufferedReader br = new BufferedReader(car);

    Class<?> cls = br.getClass();
    try {
        Field field = cls.getDeclaredField("cb");

        field.setAccessible(true);
        char[] cbVal = char[].class.cast(field.get(br));

        char[] newVal = Arrays.copyOf(cbVal, cbVal.length * 2);
        if (ACTION.equals("grow")) {
            field.set(br, newVal);
        }

        for (int i = 0; i < srcBufSize; i++) {
            br.read();
        }

        // See if the new backing array is being used
        if (newVal[srcBufSize -1] == src[srcBufSize -1]) {
            System.out.println("Using new backing array, size=" + newVal.length);
        } else {
            System.out.println("Using original backing array, size=" + cbVal.length);
        }
```

```
    } catch (NoSuchFieldException e) {
        e.printStackTrace();
    } catch (IllegalAccessException e) {
        e.printStackTrace();
    } catch (IOException e) {
        e.printStackTrace();
    }
}
```

(Oracle, 2017), "Getting and Setting Arrays and Their Components", "Setting a Field of Type Array", https://docs.oracle.com/javase/tutorial/reflect/special/arraySetGet.html
C:\Users\John\Documents\Sda\Code\Java\Examples\TutorialAtOracle\HelloWorld\src\helloworld\demo\reflection\ArrayReflection.java

Getting elements of a multidimensional array.

(Oracle, 2017), "Getting and Setting Arrays and Their Components", "Accessing Elements of a Multidimensional Array", https://docs.oracle.com/javase/tutorial/reflect/special/arraySetGet.html

Enumerated Types (Enums)

Because an Enum is a special kind of class you can use the usual reflection techniques with an Enum. However, the Enum specific techniques include the use of: `Class.isEnum()`, `Class.getEnumConstants()`, and `java.lang.reflect.Field.isEnumConstant()`.

(Oracle, 2017), "Enumerated Types", https://docs.oracle.com/javase/tutorial/reflect/special/enum.html

Get Enum Contants.

```
enum Weekday {MONDAY, TUESDAY, WEDNESDAY, THURSDAY, FRIDAY, SATURDAY, SUNDAY}

public static void start() {
    outputEnumConstants(Weekday.class);
    outputEnumConstants(java.util.concurrent.TimeUnit.class);
}

private static void outputEnumConstants(Class<?> cls) {
    String className = cls.getName();
    out.format("Enum name: %s%n" +
                   "  constants (reflection): %s%n",
            className, Arrays.asList(cls.getEnumConstants()));
    out.format("  values (non-reflection): %s%n",
            Arrays.asList(Weekday.values()));
}

// Output
Enum name: helloworld.demo.reflection.EnumeratedTypesReflection$Weekday
  constants (reflection): [MONDAY, TUESDAY, WEDNESDAY, THURSDAY, FRIDAY, SATURDAY, SUNDAY]
  values (non-reflection): [MONDAY, TUESDAY, WEDNESDAY, THURSDAY, FRIDAY, SATURDAY, SUNDAY]
Enum name: java.util.concurrent.TimeUnit
  constants (reflection): [NANOSECONDS, MICROSECONDS, MILLISECONDS, SECONDS, MINUTES, HOURS,
DAYS]
  values (non-reflection): [MONDAY, TUESDAY, WEDNESDAY, THURSDAY, FRIDAY, SATURDAY, SUNDAY]
```

(Oracle, 2017), "Examining Enums", https://docs.oracle.com/javase/tutorial/reflect/special/enumMembers.html
C:\Users\John\Documents\Sda\Code\Java\Examples\TutorialAtOracle\HelloWorld\src\helloworld\demo\reflection\EnumeratedTypesReflection.java

For an example of using the broader reflection API (in common with other classes) and `Class.isEnum()`, and `java.lang.reflect.Field.isEnumConstant()` see ...

(Oracle, 2017), "Examining Enums", https://docs.oracle.com/javase/tutorial/reflect/special/enumMembers.html

Get and set enumerated types using the same technique as for any other reference type, using `Field.get()` and `Field.set()`.

```
import java.lang.reflect.Field;
import static java.lang.System.out;

public class EnumeratedTypesGetAndSetReflection {
    enum TraceLevel {OFF, LOW, MEDIUM, HIGH, DEBUG}

    class MyServer {
        private TraceLevel traceLevel = TraceLevel.OFF;
    }

    public void setTrace(String newTraceLevelString) {
        MyServer server = new MyServer();
        Class<?> cls = server.getClass();
        TraceLevel newTraceLevel = TraceLevel.valueOf(newTraceLevelString);

        try {
            Field field = cls.getDeclaredField("traceLevel");
            field.setAccessible(true);
            TraceLevel oldTraceLevel = (TraceLevel) field.get(server);
            out.println("Original trace level: " + oldTraceLevel);

            if (oldTraceLevel != newTraceLevel) {
                field.set(server, newTraceLevel);
                out.println("   New trace level: " + field.get(server));
            }
        } catch (NoSuchFieldException e) {
            e.printStackTrace();
        } catch (IllegalAccessException e) {
            e.printStackTrace();
        }
    }
}

// In another class
...
EnumeratedTypesGetAndSetReflection etgasr = new EnumeratedTypesGetAndSetReflection();
etgasr.setTrace("DEBUG");
...

// Output
Original trace level: OFF
   New trace level: DEBUG
```

C:\Users\John\Documents\Sda\Code\Java\Examples\TutorialAtOracle\HelloWorld\src\helloworld\demo\reflection\EnumeratedTypesReflection.java
(Oracle, 2017), "Getting and Setting Fields with Enum Types",
https://docs.oracle.com/javase/tutorial/reflect/special/enumSetGet.html

Technqiues

Get the name of the current method.

```
private static void myMethod() {
    System.out.println(getCurrentMethodName());
}

private static String getCurrentMethodName() {
    return new Exception("is not thrown").getStackTrace()[1].getMethodName();
}

// Output
myMethod
```

https://stackoverflow.com/a/32714737/872154, Friso van der Made's answer to, "Getting the name of the currently executing method".

JDBC Database Access

Overview

Architecture

See (Oracle, 2017), "JDBC(TM) Database Access", https://docs.oracle.com/javase/tutorial/jdbc/index.html

With the JDBC API you do three basic tasks:

1. Connect to a datasource;
2. Send queries and update statements to the datasource.
3. Retrieve and process results.

(Oracle, 2017), "Lesson: JDBC Introduction", https://docs.oracle.com/javase/tutorial/jdbc/overview/index.html

The four JDBC components:

1. The JDBC API. Provides programmatic access to a database. Included with Java SE and Java EE. Dividided into two packages: `java.sql` and `javax.sql`.
2. The JDBC Driver Manager. Connects Java apps to the JDBC driver.
3. JDBC Test Suite. Helps you determine which JDBC drivers will run your program (?).
4. JDBC-ODBC Bridge. Provides JDBC access via ODBC drivers. Requires access to the machine to ensure the relevant ODBC binaries are installed.

(Oracle, 2017), "Lesson: JDBC Introduction", https://docs.oracle.com/javase/tutorial/jdbc/overview/index.html

The JDBC API can be used in two-tier and three-tier architectures.

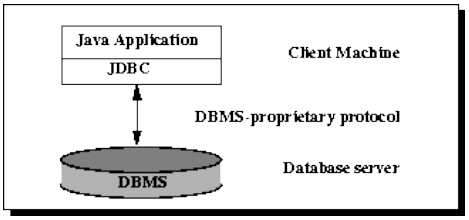


Figure 1 Two-tier architecture

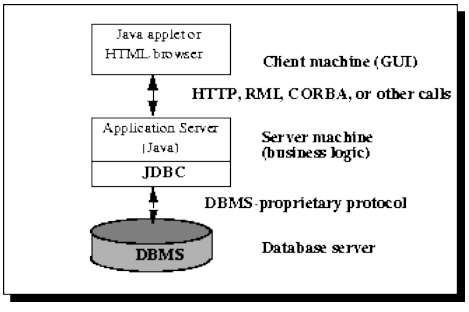


Figure 2 Three-tier architecture

(Oracle, 2017), "Lesson: JDBC Introduction", https://docs.oracle.com/javase/tutorial/jdbc/overview/index.html

JDBC Cursors, row pointers, can be used to iterate over Result Sets. Cursors can: move forward or backward; move to a specific row; or move to a row relative to the current row.

(Oracle, 2017), "Lesson: JDBC Introduction", https://docs.oracle.com/javase/tutorial/jdbc/overview/index.html

A stored procedure is a saved group of SQL statements. These are saved at the database, either:

- In DB native SQL code; or
- In Java and the JDBC API.

(Oracle, 2017), "Lesson: JDBC Introduction", https://docs.oracle.com/javase/tutorial/jdbc/overview/index.html

Setup environment

Start

See (Oracle, 2017), "JDBC(TM) Database Access", " JDBC Basics", " Getting Started", https://docs.oracle.com/javase/tutorial/jdbc/basics/gettingstarted.html

To setup your environment:

1. Install the JAVA SE SDK.
2. Install your database management system (DBMS). E.g. Java DB (included with Java SE SDK) or MySQL.
    a. For MySQL you can ensure you have the latest, if already installed by ...
        i. MySQL Notifier (the Windows taskbar program) > Right Click > Actions ... > Launch MySQLInstaller ...
        ii. [Catalog ...] (To ensure you have the latest update info]
        iii. [Upgrade ...]
3. Install a JDBC driver for your database. E.g.
    a. For JAVA DB, this comes with a JDBC driver;
    b. For MySQL, install Connector/J. https://www.mysql.com/products/connector/ (This is also available via the MySQLInstaller, see above).
4. Optionally, install the JDBC samples
    a. Install Apache Ant, a java based build tool.
        i. Download **Error! Hyperlink reference not valid.**
        ii. Decyrpt and Verify Signature file against zip file.
        iii. Install https://ant.apache.org/manual/index.html
        iv. System Environment Variables:
            1. JAVA_HOME: C:\Program Files\Java\jdk-9.0.4
            2. ANT_HOME: C:\Program Files\apache-ant-1.10.3
            3. PATH:
                a. %ANT_HOME%\bin
                b. %JAVA_HOME%\bin ... place this first in the path list, above Oracle's "C:\ProgramData\Oracle\Java\javapath"
        v. Verify from Powershell:
            1. PS> ant -version
            2. Observe a version string without errors.
    b. Download and extract the sample file, JDBCTutorial.zip.
    c. Setup the sample files. See https://docs.oracle.com/javase/tutorial/jdbc/basics/gettingstarted.html#step7
        i. In the sample java files change:

```
 "import com.sun.rowset.CachedRowSetImpl;" to "import javax.sql.rowset.*;"

//      frs = new FilteredRowSetImpl();
frs = RowSetProvider.newFactory().createFilteredRowSet();

//      jdbcRs = new JdbcRowSetImpl(con);
      jdbcRs = RowSetProvider.newFactory().createJdbcRowSet();

//      coffees = new CachedRowSetImpl();
      coffees = RowSetProvider.newFactory().createCachedRowSet();

//      suppliers = new CachedRowSetImpl();
      suppliers = RowSetProvider.newFactory().createCachedRowSet();

//      jrs = new JoinRowSetImpl();
      jrs = RowSetProvider.newFactory().createJoinRowSet();

//    WebRowSet priceList = new WebRowSetImpl();
    WebRowSet priceList = RowSetProvider.newFactory().createWebRowSet();

//    WebRowSet WebRowSetreceiver = new WebRowSetImpl();
    WebRowSet receiver = RowSetProvider.newFactory().createWebRowSet();
```

ii. In JDBCTutorial/properties/mysql-build-properties.xml

```
// Change to the following driver
<property name="DB.DRIVER" value="com.mysql.cj.jdbc.Driver" />

// Change to
    <property name="JAVAC" value="C:\\Program Files\\Java\\jdk1.8.0_201\\bin\\javac" />
    <property name="JAVA" value="C:\\Program Files\\Java\\jdk1.8.0_201\\bin\\java" />
    <property name="MYSQLDRIVER"
        value="C:\\Program Files (x86)\\MySQL\\Connector J 8.0\\mysql-connector-java-
8.0.14.jar" />

// Add "&amp;useSSL=false&amp;allowPublicKeyRetrieval=true"
    <property name="DB.URL.NEWDATABASE"
value="jdbc:mysql://${DB.HOST}:${DB.PORT}/?allowMultiQueries=true&amp;useSSL=false&amp;allow
PublicKeyRetrieval=true" />
    <property name="DB.URL"
value="jdbc:mysql://${DB.HOST}:${DB.PORT}/${DB.SID}?allowMultiQueries=true&amp;useSSL=false&
amp;allowPublicKeyRetrieval=true" />
```

iii. In JDBCTutorial/properties/mysql-sample-properties.xml

```
// Change to the following driver
<entry key="driver">com.mysql.cj.jdbc.Driver</entry>
```

iv. In com/oracle/tutorial/jdbc/JDBCTutorialUtilities.java change ...

```
  public Connection getConnection() throws SQLException {
    connectionProps.put("user", this.userName);
    connectionProps.put("password", this.password);
    // JLB Add the following ....
    connectionProps.put("useSSL", "false");
    connectionProps.put("allowPublicKeyRetrieval", "true");
```

v. Set and view global time zone:

1. Workbench Query (doesn't persist); or

```
// Set time_zone to UTC
SET GLOBAL time_zone = '+00:00';

// View
SELECT @@global.time_zone;
```

2. Workbench options file (persists).

```
// General > International >
default-time-zone: '+00:00' (with quotes)
```

vi. Run the commands from JDBCTutorial

```
ant jar  // Rerun if you change any JDBCTutorial Java classes
ant create-mysql-database
```

(Oracle, 2017), "Compile and package the samples", https://docs.oracle.com/javase/tutorial/jdbc/basics/gettingstarted.html#step9

vii. Run ant setup

```
// You should run the command ant setup every time before you run one of the Java classes in
the sample. Many of these samples expect specific data in the contents of the sample's
database tables, ... and `ant setup` repopulates the basic data.
ant setup
```

viii. Run a specific sample.

```
// Run the CoffeesTable sample
ant runct
```

If you get an error ...

> Class has been compiled by a more recent version of the Java Environment (class file version 53.0), this version of the Java Runtime only recognizes class file versions up to 52.0.

... Then ensure your JAVA_HOME environment variable is up to date (e.g. "C:\Program Files\Java\jdk-11.0.2") and match your mysql-build-properties.xml JAVA properties.

(Oracle, 2017), "Run the samples", https://docs.oracle.com/javase/tutorial/jdbc/basics/gettingstarted.html#step11

Gradle Conversion

Convert JDBCTutorial to also use gradle (in addition to ant).

1. Open JDBCTutorial in IntelliJ
2. Add the following build.gradle file to the root (JDBCTutorial\build.gradle). Ensure reference to MySQL connector is included.

```
plugins {
//    // Apply the java plugin to add support for Java
//    id 'java'
//
//    // Apply the application plugin to add support for building an application
//    id 'application'

    // Apply the java-library plugin to add support for Java Library
    id 'java-library'

}

sourceSets.main.java.srcDirs = ['src']

// Define the main class for the application
//mainClassName = 'au.com.example.myjavalibrary.Main'

dependencies {
    // This dependency is found on compile classpath of this component and consumers.
//    compile 'com.google.guava:guava:23.0'

    implementation  'mysql:mysql-connector-java:8.0.15'

    // Use JUnit test framework
```

```
//    testCompile 'junit:junit:4.12'
}

// In this section you declare where to find the dependencies of your project
repositories {
    // Use jcenter for resolving your dependencies.
    // You can declare any Maven/Ivy/file repository here.
    jcenter()
}

// https://stackoverflow.com/a/42259553/872154
task runApp(type:JavaExec) {
    classpath = sourceSets.main.runtimeClasspath
    main = project.hasProperty("mainClass") ? project.getProperty("mainClass") :
"com.oracle.tutorial.jdbc.jlb.Main"
}
```

3. Close and reopen JDBCTutorial to be prompted to import gradle changes. Go ahead and import those changes.
4. Create a gradle run configuration:
    a. Gradle Pane > JDBCTutorial > Tasks > Other > runApp. Right Click > Create "JDBCTutorial [runApp]".
    b. Run "JDBCTutorial [runApp]".

IntelliJ Tips

To copy a sql statement (e.g. from MySQL Workbench) into a java string in Intellij, paste the raw SQL into a string variable with quotes.

```
String sql = ""       // Paste inside the quotes.
```

Processing SQL statements

Overview

Processing SQL statements broadly entails:

- Handle SQLExceptions.
- Get a connection.
- Create a statement.
- Execute the query.
- Process the ResultSet object.
- Close the connection.

(Oracle, 2017), "Processing SQL Statements with JDBC", https://docs.oracle.com/javase/tutorial/jdbc/basics/processingsqlstatements.html

Handle SQLExceptions

Create some SQL error handling functions.

```
private static boolean ignoreSQLException(String sqlState) {
    if (sqlState == null) {
        System.out.println("The SQL state is not defined!");
        return false;
    }
    // X0Y32: Jar file already exists in schema
    if (sqlState.equalsIgnoreCase("X0Y32"))
```

```
            return true;
        // 42Y55: Table already exists in schema
        if (sqlState.equalsIgnoreCase("42Y55"))
            return true;
        return false;
    }

    public static void handleSQLException(SQLException ex) {
        for (Throwable e : ex) {
            if (e instanceof SQLException) {
                if (ignoreSQLException(((SQLException) e).getSQLState()) == false) {
                    switch (((SQLException) e).getSQLState()) {
                        case "08S01":
                            // Communications link failure
                            System.err.println("MySQL Server may be down. Please start it");
                            outputSQLException(ex, e);
                            break;
                        default:
                            e.printStackTrace(System.err);
                            outputSQLException(ex, e);
                    }
                }
            }
        }
    }

    private static void outputSQLException(SQLException ex, Throwable e) {
        System.err.println("SQLState: " + ((SQLException) e).getSQLState());
        System.err.println("Error Code: " + ((SQLException) e).getErrorCode());
        System.err.println("Message: " + e.getMessage());
        Throwable t = ex.getCause();
        while (t != null) {
            System.out.println("Cause: " + t);
            t = t.getCause();
        }
    }
```

From (Oracle, 2017), com/oracle/tutorial/jdbc/JDBCTutorialUtilities.java, at "JDBC Basics", "Getting Started", https://docs.oracle.com/javase/tutorial/jdbc/basics/gettingstarted.html
C:\Users\John\Documents\Sda\Code\Java\Examples\JDBCTutorial\src\com\oracle\tutorial\jdbc\jlb\DBConnections.java

Also handle warnings.

```
public class DBConnections {
...
    public static void getWarningsFromResultSet(ResultSet rs) throws SQLException {
        JDBCTutorialUtilities.printWarnings(rs.getWarnings());
    }

    public static void getWarningsFromStatement(Statement stmt) throws SQLException {
        JDBCTutorialUtilities.printWarnings(stmt.getWarnings());
    }

    public static void getWarningsFromConnection(Connection connection) throws SQLException
{
        JDBCTutorialUtilities.printWarnings(connection.getWarnings());
    }

    public static void printWarnings(SQLWarning warning) throws SQLException {
        if (warning != null) {
            System.out.println("\n---Warning---\n");
            while (warning != null) {
                System.out.println("Message: " + warning.getMessage());
                System.out.println("SQLState: " + warning.getSQLState());
                System.out.print("Vendor error code: ");
                System.out.println(warning.getErrorCode());
                System.out.println("");
                warning = warning.getNextWarning();
            }
        }
    }
}
```

```
...

statement = connection.createStatement();
ResultSet resultSet = statement.executeQuery(query);
DBConnections.getWarningsFromStatement(statement);
DBConnections.getWarningsFromResultSet(resultSet);
```

C:\Users\John\Documents\Sda\Code\Java\Examples\TutorialAtOracle\JDBCTutorial\src\jlb\DBConnections.java
C:\Users\John\Documents\Sda\Code\Java\Examples\TutorialAtOracle\JDBCTutorial\src\com\oracle\tutorial\jdbc\JDBCTutorialUtilities.java
(Oracle, 2017), "Retrieving Warnings",
https://docs.oracle.com/javase/tutorial/jdbc/basics/sqlexception.html#retrieving_warnings

Directly handle SQLExceptions as they occur.

```
        } catch (SQLException e) {
            // Table 'X' already exists
            if (e.getSQLState().equalsIgnoreCase("42S01")){
                System.err.println(e.getMessage());
            } else {
                DBConnections.printSQLException(e);
            }
        }
```

C:\Users\John\Documents\Sda\Code\Java\Examples\TutorialAtOracle\JDBCTutorial\src\jlb\CreateAndPopulateTables.java

Get a connection (MySQL)

Setup

Verify you have "JDBC Driver for MySQL (Connector/J)" installed and the latest version:

- {MySQL Notifier (Tray program)} > Right Click > Actions > Launch MySQL installer ...
- [Catalog ...] > [Execute]. This updates the catalog (of available versions); [Next] > [Finsih]
- If necessary [Upgrade].
- Observe the existence of "C:\Program Files (x86)\MySQL\Connector J 8.0\mysql-connector-java-8.0.11.jar".

https://www.mysql.com/products/connector/
(Oracle, 2017), "Install a JDBC driver from the vendor of your database",
https://docs.oracle.com/javase/tutorial/jdbc/basics/gettingstarted.html#step3

Set the "JDBC Driver for MySQL (Connector/J)" as a dependency of your project:

1. Natively

   E.g. Add C:\Program Files (x86)\MySQL\Connector J 8.0\mysql-connector-java-8.0.11.jar as a dependency.

   (Bentley, Java Reference - Language.docx, 2017), "IDE dependency setting in lieu of adding a CLASSPATH"

2. Via gradle

```
dependencies {
    // This dependency is found on compile classpath of this component and consumers.
//    compile 'com.google.guava:guava:23.0'

    implementation  'mysql:mysql-connector-java:8.0.15'
```

```
    implementation  project(':sjl')

    // Use JUnit test framework
//    testCompile 'junit:junit:4.12'
}
```

C:\Users\John\Documents\Sda\Code\Java\Examples\JDBCTutorial\build.gradle

Overview

You connect to a datasource using either a `DriverManager` or `DataSource` class:

- `DriverManager` class: uses a URL to connect; loads any JDBC 4.0 drivers found in the CLASSPATH (or otherwise referenced via a dependency).
- (Recommended) `DataSource` class. Preferred because: it allows details about the underlying data source to be transparent to your application ("Establishing a Connection"); provices connection pooling and distributed transactions ("Connecting with DataSource Objects").

(Oracle, 2017), "Establishing a Connection", https://docs.oracle.com/javase/tutorial/jdbc/basics/connecting.html
(Oracle, 2017), "Connecting with DataSource Objects", https://docs.oracle.com/javase/tutorial/jdbc/basics/sqldatasources.html

DriverManager

Get a connection, using a `DriverManager`.

```
public class DBConnections {

    static private final String serverName = "localhost";
    static private final String portNumber = "3306";
    static private final String user = "root";
    static private final String password = "somepassword";

    static public Connection getConnectionUsingDriverManager(String database) throws
SQLException {
        final String dbms = "mysql";

        Connection connection = null;
        Properties connectionProperties = new Properties();
        connectionProperties.put("user", user);
        connectionProperties.put("password", password);
        connectionProperties.put("useSSL", "false");


        // MySQL connection
        String connectionString = "jdbc:" +
                dbms + "://" +
                serverName + ":" +
                portNumber + "/" +
                database;
        try {
            // Smack the driver in the head, otherwise none of this works
            Class<?> cls = Class.forName("com.mysql.cj.jdbc.Driver");
            cls.getDeclaredConstructor().newInstance();

            connection = DriverManager.getConnection(connectionString,
connectionProperties);
            System.out.println("Connected: " + connectionString);

        } catch (ClassNotFoundException | IllegalAccessException
                | NoSuchMethodException | InvocationTargetException
                | InstantiationException e) {
            e.printStackTrace();
        }
        return connection;
    }
```

C:\Users\John\Documents\Sda\Code\Java\Examples\TutorialAtOracle\JDBCTutorial\src\jlb\DBConnections.java
(Oracle, 2017), "Establishing a Connection", https://docs.oracle.com/javase/tutorial/jdbc/basics/connecting.html

Using a connection ...

```
public static void main(String[] args) {
    final String database = "testdb";

    try (Connection connection = getConnection(database);) {
        viewTable(connection, database);
    } catch (SQLException e) {
        printSQLException(e);
    }
}

...

    static public void viewTable(Connection connection, String database) throws SQLException
{
        String query = "select COF_NAME, SUP_ID, PRICE, SALES, TOTAL " +
                "from " + database + ".COFFEES;";

        String coffeeName = "";
        int supplierID = 0;
        float price = 0f;
        int sales = 0;
        int total = 0;

        try (Statement statement = connection.createStatement();) {

            ResultSet resultSet = statement.executeQuery(query);

            while (resultSet.next()) {
                coffeeName = resultSet.getString("COF_NAME");
                supplierID = resultSet.getInt("SUP_ID");
                price = resultSet.getFloat("PRICE");
                sales = resultSet.getInt("SALES");
                total = resultSet.getInt("TOTAL");

                System.out.printf("%-20s %4d %6.2f %4d %4d%n", coffeeName, supplierID,
price, sales, total);
            }

        } catch (SQLException e) {
            e.printStackTrace();
        }
    }
}
```

C:\Users\John\Documents\Sda\Code\Java\Examples\TutorialAtOracle\JDBCTutorial\src\com\oracle\tutorial\jdbc\JlbMain.java
(Oracle, 2017), "Processing SQL Statements with JDBC",
https://docs.oracle.com/javase/tutorial/jdbc/basics/processingsqlstatements.html

DataSource

The better kind of connection, because it allows connection pooling (among other things). However, potentially complex to setup.

Create a statement

bookmark

Overview

A `statement` is an interface which represents a SQL statement. They generally return `ResultSet` objects. There are three kind of statement objects:

- A plain `Statement`. For SQL statements without paramters.
- `PreparedStatement` (A subclass of `Statement`). For precompiling SQL statements that might contain input parameters.
- `CalleableStatement` (A subclass of `PreparedStatement`). For executing SQL stored procedures that may contain both input and output parameters.

```
        try (Statement statement = connection.createStatement();
             ...) {
```

(Oracle, 2017), "Processing SQL Statements with JDBC", "Creating Statements"
https://docs.oracle.com/javase/tutorial/jdbc/basics/processingsqlstatements.html
C:\Users\John\Documents\Sda\Code\Java\Examples\TutorialAtOracle\JDBCTutorial\src\jlb\JlbMain.java

Kinds of Statements

Plain Statement

The common case, as above.

Prepared Statement

PreparedStatements can accept parameters, via a 1 based index.

```
public class TableSuppliers {
    int SUP_ID = 0;
    String SUP_NAME =  "";
    String STREET = "";
    String CITY = "";
    String STATE = "";
    String ZIP = "";

    public TableSuppliers(int SUP ID, String SUP NAME, String STREET, String CITY, String
STATE, String ZIP) {
        this.SUP_ID = SUP_ID;
        this.SUP_NAME = SUP_NAME;
        this.STREET = STREET;
        this.CITY = CITY;
        this.STATE = STATE;
        this.ZIP = ZIP;
    }
}

static TableSuppliers[] dataRows = {
        new TableSuppliers(1000, "Superior Coffee", "1 Party Place", "Mendocino", "CA",
"95460"),
        new TableSuppliers(1001, "Very Bi. Coffee", "5 Party Place", "Ventura", "CA",
"93009")
};

static public void InsertViaSqlUsingPreparedStatement (Connection connection,
                                                       String databaseName,
                                                       String tableName) {
    String tableNameQualified = databaseName + '.' + tableName;
    String sql = "";

    sql = "INSERT INTO " + tableNameQualified + " VALUES (";
    sql += "?, ?, ?, ?, ?, ?);";

    try (PreparedStatement preparedStatement = connection.prepareStatement(sql)) {

        connection.setAutoCommit(false);

        for (TableSuppliers supplier : dataRows) {
```

```
            // Set parameters via a 1 based index
            preparedStatement.setInt(1, supplier.SUP_ID);
            preparedStatement.setString(2, supplier.SUP_NAME);
            preparedStatement.setString(3, supplier.STREET);
            preparedStatement.setString(4, supplier.CITY);
            preparedStatement.setString(5, supplier.STATE);
            preparedStatement.setString(6, supplier.ZIP);

            System.out.println(sql);

            // Don't supply sql
            preparedStatement.executeUpdate();
            DBConnections.getWarningsFromStatement(preparedStatement);
        }

        connection.commit();
        connection.setAutoCommit(true);

        System.out.printf("PreparedStatement (paramaterized) update insertion into: %s%n",
tableNameQualified);


    } catch (SQLException e) {
        if (connection != null) {
            try {
                connection.rollback();
                System.err.println("Rolledback update attempt.");
            } catch (SQLException rolle) {
                DBConnections.handleSQLException(rolle);
            }
        }
        // Duplicate entry 'X' for key 'Y'
        if (e.getSQLState().equalsIgnoreCase("23000")) {
            System.err.println(e.getMessage());
        } else {
            DBConnections.handleSQLException(e);
        }
    }
}
```

C:\Users\John\Documents\Sda\Code\Java\Examples\TutorialAtOracle\JDBCTutorial\src\jlb\DataManipulationOperations.java
(Oracle, 2017), " Using Prepared Statements", https://docs.oracle.com/javase/tutorial/jdbc/basics/prepared.html

PreparedStatements can also be used in batch updates. See also: Batch updates.

```
static TableSuppliers[] dataRows = {
        new TableSuppliers(1000, "Superior Coffee", "1 Party Place", "Mendocino", "CA",
"95460"),
        new TableSuppliers(1001, "Very Bi. Coffee", "5 Party Place", "Ventura", "CA",
"93009")
};


static public void InsertViaSqlInsideBatchUpdateUsingPreparedStatement(Connection
connection,
                                                                       String databaseName,
                                                                       String tableName) {
    String tableNameQualified = databaseName + '.' + tableName;
    String sql = "";

    sql = "INSERT INTO " + tableNameQualified + " VALUES (";
    sql += "?, ?, ?, ?, ?, ?);";

    try (PreparedStatement preparedStatement = connection.prepareStatement(sql)) {

        connection.setAutoCommit(false);

        for (TableSuppliers supplier : dataRows) {

            // Set parameters via a 1 based index
            preparedStatement.setInt(1, supplier.SUP_ID);
```

```
            preparedStatement.setString(2, supplier.SUP_NAME);
            preparedStatement.setString(3, supplier.STREET);
            preparedStatement.setString(4, supplier.CITY);
            preparedStatement.setString(5, supplier.STATE);
            preparedStatement.setString(6, supplier.ZIP);

            System.out.println(sql);

            // Don't supply sql
            preparedStatement.addBatch();
            DBConnections.getWarningsFromStatement(preparedStatement);
        }

        int[] updateCounts = preparedStatement.executeBatch();
        connection.commit();

        System.out.printf("Batch update insertion into: %s%n", tableNameQualified);
        System.out.printf("updateCounts Array: %s%n", Arrays.toString(updateCounts));

    } catch (SQLException e) {
        // Duplicate entry 'X' for key 'Y'
        if (e.getSQLState().equalsIgnoreCase("23000")) {
            System.err.println(e.getMessage());
        } else {
            DBConnections.handleSQLException(e);
        }
    }
}
```

C:\Users\John\Documents\Sda\Code\Java\Examples\TutorialAtOracle\JDBCTutorial\src\jlb\DataManipulationOperations.java
(Oracle, 2017), "Retrieving and Modifying Values from Result Sets", "Performing Parameterized Batch Update"
https://docs.oracle.com/javase/tutorial/jdbc/basics/retrieving.html#batch_updates

Callable Statement

Execute the query

To execute a query use an `execute*` method on a `Statement` object. There are three types of `execute*` methods:

- `execute`. Returns true if the first object is a `ResultSet`. Use this if the query will return one or more `ResultSet` objects.
- `excecuteQuery`. Return only one `ResultSet`.
- `executeUpdate`. Returns integer representing the number of rows effected. Use for an INSERT, DELETE, or UPDATE SQL statement.

```
    static public void viewTable(Connection connection, String database) throws SQLException
{
        String query = "select COF_NAME, SUP_ID, PRICE, SALES, TOTAL " +
                "from " + database + ".COFFEES;";

...

        try (Statement statement = connection.createStatement();
             ResultSet resultSet = statement.executeQuery(query)) {
```

(Oracle, 2017), "Processing SQL Statements with JDBC", "Executing Queries",
https://docs.oracle.com/javase/tutorial/jdbc/basics/processingsqlstatements.html#executing_queries
C:\Users\John\Documents\Sda\Code\Java\Examples\TutorialAtOracle\JDBCTutorial\src\jlb\JlbMain.java

Process the ResultSet object

Process a `ResultSet` object with a cursor ("not a database cursor", whatever that entails). A cursor is a row pointer. Initially it points to the first row.

```
        try (Statement statement = connection.createStatement()) {
            ResultSet resultSet = statement.executeQuery(query));

            while (resultSet.next()) {
                coffeeName = resultSet.getString("COF_NAME");
                supplierID = resultSet.getInt("SUP_ID");
                price = resultSet.getFloat("PRICE");
                sales = resultSet.getInt("SALES");
                total = resultSet.getInt("TOTAL");

                System.out.printf("%-20s %4d %6.2f %4d %4d%n", coffeeName, supplierID,
price, sales, total);
            }
```

(Oracle, 2017), "Processing SQL Statements with JDBC", "Processing Result Sets",
https://docs.oracle.com/javase/tutorial/jdbc/basics/processingsqlstatements.html#processing_resultset_objects
C:\Users\John\Documents\Sda\Code\Java\Examples\TutorialAtOracle\JDBCTutorial\src\jlb\JlbMain.java

Close the connection

You can close a Statement object either:

- Classically; or
- (Recommended) Using a try-with-resources declaration (which will then perform the closing as needed).

Closing a statement will also close the ResultSet (so you don't need to worry about closing ResultSets yourself).

(Oracle, 2017), "Processing SQL Statements with JDBC", " Closing Connections",
https://docs.oracle.com/javase/tutorial/jdbc/basics/processingsqlstatements.html#closing_connections
C:\Users\John\Documents\Sda\Code\Java\Examples\TutorialAtOracle\JDBCTutorial\src\jlb\JlbMain.java

Close a statement (and so ResultSet) classically.

```
// Declare statement outside of try block so it can be referenced in finally block
Statement statement = null;
try  {
    statement = connection.createStatement();
    ResultSet resultSet = statement.executeQuery(query);

...

} catch (SQLException e) {
    e.printStackTrace();
} finally {
    if (statement != null) { statement.close(); }
}
```

(Oracle, 2017), "Processing SQL Statements with JDBC", " Closing Connections",
https://docs.oracle.com/javase/tutorial/jdbc/basics/processingsqlstatements.html#closing_connections
C:\Users\John\Documents\Sda\Code\Java\Examples\TutorialAtOracle\JDBCTutorial\src\jlb\JlbMain.java

Use a try-with-resources statement to automatically close Connection and Statement (and so ResultSet) objects.

```
public static void main(String[] args) {
    final String database = "testdb";

    try (Connection connection = getConnection(database);) {
        viewTable(connection, database);
    } catch (SQLException e) {
        printSQLException(e);
    }
}
```

```
...

    static public void viewTable(Connection connection, String database) throws SQLException
{
        ...

        try (Statement statement = connection.createStatement();) {

            ResultSet resultSet = statement.executeQuery(query);

            while (resultSet.next()) {
                ....
            }

        } catch (SQLException e) {
            e.printStackTrace();
        }
    }
}
```

(Oracle, 2017), "Processing SQL Statements with JDBC", " Closing Connections",
https://docs.oracle.com/javase/tutorial/jdbc/basics/processingsqlstatements.html#closing_connections
C:\Users\John\Documents\Sda\Code\Java\Examples\TutorialAtOracle\JDBCTutorial\src\jlb\JlbMain.java

Full Example

```
public class DBConnections {

    static private final String serverName = "localhost";
    static private final String portNumber = "3306";
    static private final String user = "root";
    static private final String password = "somepassword";

    static public Connection getConnectionUsingDriverManager(String database) throws
SQLException {
        final String dbms = "mysql";

        Connection connection = null;
        Properties connectionProperties = new Properties();
        connectionProperties.put("user", user);
        connectionProperties.put("password", password);
        connectionProperties.put("useSSL", "false");


        // MySQL connection
        String connectionString = "jdbc:" +
                dbms + "://" +
                serverName + ":" +
                portNumber + "/" +
                database;
        try {
            // Smack the driver in the head, otherwise none of this works
            Class<?> cls = Class.forName("com.mysql.cj.jdbc.Driver");
            cls.getDeclaredConstructor().newInstance();

            connection = DriverManager.getConnection(connectionString,
connectionProperties);
            System.out.println("Connected: " + connectionString);

        } catch (ClassNotFoundException | IllegalAccessException
                | NoSuchMethodException | InvocationTargetException
                | InstantiationException e) {
            e.printStackTrace();
        }
        return connection;
    }

    private static boolean ignoreSQLException(String sqlState) {
        if (sqlState == null) {
            System.out.println("The SQL state is not defined!");
            return false;
        }
        // X0Y32: Jar file already exists in schema
        if (sqlState.equalsIgnoreCase("X0Y32"))
```

```
            return true;
        // 42Y55: Table already exists in schema
        if (sqlState.equalsIgnoreCase("42Y55"))
            return true;
        return false;
    }

    public static void printSQLException(SQLException ex) {
        for (Throwable e : ex) {
            if (e instanceof SQLException) {
                if (ignoreSQLException(((SQLException) e).getSQLState()) == false) {
                    e.printStackTrace(System.err);
                    System.err.println("SQLState: " + ((SQLException) e).getSQLState());
                    System.err.println("Error Code: " + ((SQLException) e).getErrorCode());
                    System.err.println("Message: " + e.getMessage());
                    Throwable t = ex.getCause();
                    while (t != null) {
                        System.out.println("Cause: " + t);
                        t = t.getCause();
                    }
                }
            }
        }
    }
}
```

C:\Users\John\Documents\Sda\Code\Java\Examples\TutorialAtOracle\JDBCTutorial\src\jlb\DBConnections.java

```
public class JlbMain {
    public static void main(String[] args) {
        startDatabaseDemo();
    }

    public static void startDatabaseDemo() {
        final String database = "testdb";

        try (Connection connection =
DBConnections.getConnectionUsingDriverManager(database);) {
//        try (Connection connection =
DBConnections.getConnectionUsingDataSourceStandard(database);) {
            viewTable(connection, database);
        } catch (SQLException e) {
            DBConnections.printSQLException(e);
        }
    }


    static public void viewTable(Connection connection, String database) throws SQLException
{
        String query = "select COF_NAME, SUP_ID, PRICE, SALES, TOTAL " +
                "from " + database + ".COFFEES;";

        String coffeeName = "";
        int supplierID = 0;
        float price = 0f;
        int sales = 0;
        int total = 0;

        try (Statement statement = connection.createStatement()) {

            ResultSet resultSet = statement.executeQuery(query);
            DBConnections.getWarningsFromStatement(statement);
            DBConnections.getWarningsFromResultSet(resultSet);

            while (resultSet.next()) {
                coffeeName = resultSet.getString("COF_NAME");
                supplierID = resultSet.getInt("SUP_ID");
                price = resultSet.getFloat("PRICE");
                sales = resultSet.getInt("SALES");
                total = resultSet.getInt("TOTAL");

                System.out.printf("%-20s %4d %6.2f %4d %4d%n", coffeeName, supplierID,
price, sales, total);
            }
```

```
        } catch (SQLException e) {
            DBConnections.printSQLException(e);
        }
    }
}
```

C:\Users\John\Documents\Sda\Code\Java\Examples\TutorialAtOracle\JDBCTutorial\src\jlb\JlbMain.java

Data Definition Operations

Create tables

Create a table with an `executeUpdate` on a `statement` object.

```
// We have a previously created connection object.

populateTable(connection, databaseName, "jlbSuppliers", dataRows);

static public void createTable(Connection connection,
                               String databaseName,
                               String tableName) {
    String tableNameQualified = databaseName + '.' + tableName;
    String sql = "CREATE TABLE " + tableNameQualified + "(\n" +
            "  SUP_ID int(11) NOT NULL,\n" +
            "  SUP_NAME varchar(40) NOT NULL,\n" +
            "  STREET varchar(40) NOT NULL,\n" +
            "  CITY varchar(20) NOT NULL,\n" +
            "  STATE char(2) NOT NULL,\n" +
            "  ZIP char(5) DEFAULT NULL,\n" +
            "  PRIMARY KEY (SUP_ID)\n" +
            ") ENGINE=InnoDB DEFAULT CHARSET=utf8";


    try (Statement statement = connection.createStatement()) {

        int rowsAffected = statement.executeUpdate(sql);
        DBConnections.getWarningsFromStatement(statement);
        System.out.printf("Created: %s%n%n", tableNameQualified);

    } catch (SQLException e) {
        // Table 'X' already exists
        if (e.getSQLState().equalsIgnoreCase("42S01")) {
            System.err.println(e.getMessage());
        } else {
            DBConnections.handleSQLException(e);
        }
    }
}

// Output
Created: testdb.jlbSuppliers

// Observe MySql server table jlbSuppliers
```

C:\Users\John\Documents\Sda\Code\Java\Examples\TutorialAtOracle\JDBCTutorial\src\jlb\CreateAndPopulateTables.java
(Oracle, 2017), "Setting Up Tables", "Creating Tables",
https://docs.oracle.com/javase/tutorial/jdbc/basics/tables.html#create

Data Manipulation Operations

Insert values

Via SQL insert

To populate a table:

- Create a class that mimicks the table structure (e.g. TableSuppliers);

```
public class TableSuppliers {
    int SUP_ID = 0;
    String SUP_NAME =  "";
    String STREET = "";
    String CITY = "";
    String STATE = "";
    String ZIP = "";

    public TableSuppliers(int SUP_ID, String SUP_NAME, String STREET, String CITY,
String STATE, String ZIP) {
        this.SUP_ID = SUP_ID;
        this.SUP_NAME = SUP_NAME;
        this.STREET = STREET;
        this.CITY = CITY;
        this.STATE = STATE;
        this.ZIP = ZIP;
    }
}
```

C:\Users\John\Documents\Sda\Code\Java\Examples\TutorialAtOracle\JDBCTutorial\src\jlb\TableSuppliers.java

- Create a dataRows array of instances of the class.

```
    static TableSuppliers[] dataRows = {
            new TableSuppliers(1000, "Superior Coffee", "1 Party Place", "Mendocino",
"CA", "95460"),
            new TableSuppliers(1001, "Very Bi. Coffee", "5 Party Place", "Ventura",
"CA", "93009")
    };
```

C:\Users\John\Documents\Sda\Code\Java\Examples\TutorialAtOracle\JDBCTutorial\src\jlb\DataManipulationOperations.java

- Loop through the dataRows array to create an INSERT INTO sql string; and Use `statement.executeUpdate(sql).`

```
    static public void populateTable(Connection connection,
                                     String databaseName,
                                     String tableName,
                                     TableSuppliers[] dataRows) {
        String tableNameQualified = databaseName + '.' + tableName;
        String sql = "";

        try (Statement statement = connection.createStatement()) {
            for (TableSuppliers supplier : dataRows) {
                sql = "INSERT INTO " + tableNameQualified + " VALUES (";
                sql += supplier.SUP_ID + ", ";
                sql += String.format("'%s', ", supplier.SUP_NAME);
                sql += String.format("'%s', ", supplier.STREET);
                sql += String.format("'%s', ", supplier.CITY);
                sql += String.format("'%s', ", supplier.STATE);
                sql += String.format("'%s'", supplier.ZIP);
                sql += ");";
                System.out.println(sql);
                int rowsAffected = statement.executeUpdate(sql);
                DBConnections.getWarningsFromStatement(statement);
```

```
                System.out.printf("Inserted into: %s%n", tableNameQualified);
                System.out.printf("rowsAffected: %d%n%n", rowsAffected);
            }
        } catch (SQLException e) {
            // Duplicate entry 'X' for key 'Y'
            if (e.getSQLState().equalsIgnoreCase("23000")) {
                System.err.println(e.getMessage());
            } else {
                DBConnections.handleSQLException(e);
            }
        }
    }
```

C:\Users\John\Documents\Sda\Code\Java\Examples\TutorialAtOracle\JDBCTutorial\src\jlb\DataManipulationOperations.java
(Oracle, 2017), "Setting Up Tables", " Populating Tables",
https://docs.oracle.com/javase/tutorial/jdbc/basics/tables.html#populate

Via ResultSet

Insert a row via a ResultSet as follows:

```
static public void InsertViaResultSet(Connection connection,
                                      String databaseName) {

    String tableNameQualified = databaseName + ".COFFEES";
    String sql = "";

    try (Statement statement = connection.createStatement(ResultSet.TYPE_SCROLL_SENSITIVE,
ResultSet.CONCUR_UPDATABLE)) {

        ResultSet resultSet = statement.executeQuery("SELECT * FROM " + tableNameQualified);
        DBConnections.getWarningsFromStatement(statement);
        DBConnections.getWarningsFromResultSet(resultSet);

        resultSet.moveToInsertRow();

        resultSet.updateString("COF_NAME", "Great Blend");
        resultSet.updateInt("SUP_ID", 150);
        resultSet.updateFloat("PRICE", 11.11f);
        resultSet.updateInt("SALES", 66);
        resultSet.updateInt("TOTAL", 66);

        resultSet.insertRow();
        resultSet.beforeFirst();

        System.out.format("Inserted a row via ResultSet to %s%n", tableNameQualified);
    } catch (SQLException e) {
        DBConnections.handleSQLException(e);
    }
}
```

C:\Users\John\Documents\Sda\Code\Java\Examples\TutorialAtOracle\JDBCTutorial\src\jlb\DataManipulationOperations.java
(Oracle, 2017), "Retrieving and Modifying Values from Result Sets", "Inserting Rows in ResultSet Objects",
https://docs.oracle.com/javase/tutorial/jdbc/basics/retrieving.html#rs_insert

Modify values

About ResultSets

A `ResultSet` object has three important characteristics:

- Type: TYPE_FORWARD_ONLY; TYPE_SCROLL_INSENSTIVE (insensitive to changes maded in underlying data source while open); TYPE_SCROLL_SENSITIVE.
- Concurrency: CONCUR_READ_ONLY; CONCUR_UPDATEABLE.
- Cursor Holdability: HOLD_CURSORS_OVER_COMMIT (hold cursor open after commit); CLOSE_CURSORS_AT_COMMIT.

(Oracle, 2017), "Retrieving and Modifying Values from Result Sets", "ResultSet Interface", https://docs.oracle.com/javase/tutorial/jdbc/basics/retrieving.html#rs_interface

To retrieve column values with a ResultSet use either: Column or Column alias name; or numerical index (1 based).

```
// By Name
price = resultSet.getFloat("PRICE");

// By Index
price = resultSet.getFloat(3);
```

(Oracle, 2017), "Retrieving and Modifying Values from Result Sets", "Retrieving Column Values from Rows", https://docs.oracle.com/javase/tutorial/jdbc/basics/retrieving.html#retrieve_rs

ResultSet cursor movement commands:

- next; previous;
- first; last;
- beforeFirst; afterLast;
- relative(int rows); absolute (int row)

(Oracle, 2017), "Retrieving and Modifying Values from Result Sets", " Cursors", https://docs.oracle.com/javase/tutorial/jdbc/basics/retrieving.html#cursors

Update with a ResultSet

To update values via a ResultSet: set concurrence to updateable; use an update* method; and updateRow() method.

```
static public void UpdateWithResultSet(Connection connection,
                                        String databaseName,
                                        String tableName) {

    String tableNameQualified = databaseName + '.' + tableName;
    String sql = "";

    try (Statement statement = connection.createStatement(ResultSet.TYPE_SCROLL_SENSITIVE,
ResultSet.CONCUR_UPDATABLE)) {

        ResultSet resultSet = statement.executeQuery("SELECT * FROM " + tableNameQualified);
        DBConnections.getWarningsFromStatement(statement);
        DBConnections.getWarningsFromResultSet(resultSet);

        float percentageIncrease = 1.1f;
        while (resultSet.next()) {
            float oldPrice = resultSet.getFloat("PRICE");
            float newPrice = oldPrice * percentageIncrease;
            resultSet.updateFloat("PRICE",  newPrice);

            // Necessary to effect a save.
            resultSet.updateRow();
            System.out.format("old: %f  new: %f%n", oldPrice, newPrice);
        }

    } catch (SQLException e) {
```

```
            DBConnections.handleSQLException(e);
        }
    }
```

(Oracle, 2017), "Retrieving and Modifying Values from Result Sets", "Updating Rows in ResultSet Objects", https://docs.oracle.com/javase/tutorial/jdbc/basics/retrieving.html#rs_update
C:\Users\John\Documents\Sda\Code\Java\Examples\TutorialAtOracle\JDBCTutorial\src\jlb\ModifyValues.java

Batch updates

For `Statement`, `PreparedStatement`, and `CallableStatement` objects a list of data manipulation or data definition SQL statements can be associated with them. That is, any SQL statement that does not return a ResultSet (as a SQL SELECT would). These can be then executed in a batch. Do this as follows:

```
static TableSuppliers[] dataRows = {
        new TableSuppliers(1000, "Superior Coffee", "1 Party Place", "Mendocino", "CA",
"95460"),
        new TableSuppliers(1001, "Very Bi. Coffee", "5 Party Place", "Ventura", "CA",
"93009")
};

static public void InsertViaSqlInsideBatchUpdate(Connection connection,
                                                 String databaseName,
                                                 String tableName) {
    String tableNameQualified = databaseName + '.' + tableName;
    String sql = "";

    try (Statement statement = connection.createStatement()) {

        connection.setAutoCommit(false);

        for (TableSuppliers supplier : dataRows) {
            sql = "INSERT INTO " + tableNameQualified + " VALUES (";
            sql += supplier.SUP_ID + ", ";
            sql += String.format("'%s', ", supplier.SUP_NAME);
            sql += String.format("'%s', ", supplier.STREET);
            sql += String.format("'%s', ", supplier.CITY);
            sql += String.format("'%s', ", supplier.STATE);
            sql += String.format("'%s'", supplier.ZIP);
            sql += ");";
            System.out.println(sql);
            statement.addBatch(sql);
            DBConnections.getWarningsFromStatement(statement);
        }

        int[] updateCounts = statement.executeBatch();
        connection.commit();

        System.out.printf("Batch update insertion into: %s%n", tableNameQualified);
        System.out.printf("updateCounts Array: %s%n", Arrays.toString(updateCounts));

        connection.setAutoCommit(true);

    } catch (SQLException e) {
        // Duplicate entry 'X' for key 'Y'
        if (e.getSQLState().equalsIgnoreCase("23000")) {
            System.err.println(e.getMessage());
        } else {
            DBConnections.handleSQLException(e);
        }
    }
}
```

C:\Users\John\Documents\Sda\Code\Java\Examples\TutorialAtOracle\JDBCTutorial\src\jlb\DataManipulationOperations.java
(Oracle, 2017), "Retrieving and Modifying Values from Result Sets", "Using Statement Objects for Batch Updates", https://docs.oracle.com/javase/tutorial/jdbc/basics/retrieving.html#batch_updates

Batch update via a Parameterized SQL statement, using a `PreparedStatement`.

```
static public void InsertViaSqlInsideBatchUpdateUsingPreparedStatement(Connection
connection,
                                                                       String databaseName,
                                                                       String tableName) {
    String tableNameQualified = databaseName + '.' + tableName;
    String sql = "";

    sql = "INSERT INTO " + tableNameQualified + " VALUES (";
    sql += "?, ?, ?, ?, ?, ?);";

    try (PreparedStatement preparedStatement = connection.prepareStatement(sql)) {

        connection.setAutoCommit(false);

        for (TableSuppliers supplier : dataRows) {

            // Set parameters via a 1 based index
            preparedStatement.setInt(1, supplier.SUP_ID);
            preparedStatement.setString(2, supplier.SUP_NAME);
            preparedStatement.setString(3, supplier.STREET);
            preparedStatement.setString(4, supplier.CITY);
            preparedStatement.setString(5, supplier.STATE);
            preparedStatement.setString(6, supplier.ZIP);

            System.out.println(sql);

            // Don't supply sql
            preparedStatement.addBatch();
            DBConnections.getWarningsFromStatement(preparedStatement);
        }

        int[] updateCounts = preparedStatement.executeBatch();
        connection.commit();

        connection.setAutoCommit(true);

        System.out.printf("Batch update insertion into: %s%n", tableNameQualified);
        System.out.printf("updateCounts Array: %s%n", Arrays.toString(updateCounts));

    } catch (SQLException e) {
        // Duplicate entry 'X' for key 'Y'
        if (e.getSQLState().equalsIgnoreCase("23000")) {
            System.err.println(e.getMessage());
        } else {
            DBConnections.handleSQLException(e);
        }
    }
}
```

C:\Users\John\Documents\Sda\Code\Java\Examples\TutorialAtOracle\JDBCTutorial\src\jlb\DataManipulationOperations.java
(Oracle, 2017), "Retrieving and Modifying Values from Result Sets", "Using Statement Objects for Batch Updates", https://docs.oracle.com/javase/tutorial/jdbc/basics/retrieving.html#batch_updates

Transactions

Transactions allow several SQL statements to be excucted at as a group, or rolled back as a group.

```
static public void InsertViaSqlUsingPreparedStatement (Connection connection,
                                                       String databaseName,
                                                       String tableName) {
    String tableNameQualified = databaseName + '.' + tableName;
    String sql = "";

    sql = "INSERT INTO " + tableNameQualified + " VALUES (";
    sql += "?, ?, ?, ?, ?, ?);";
```

```
    try (PreparedStatement preparedStatement = connection.prepareStatement(sql)) {

        connection.setAutoCommit(false);

        for (TableSuppliers supplier : dataRows) {

            // Set parameters via a 1 based index
            preparedStatement.setInt(1, supplier.SUP_ID);
            preparedStatement.setString(2, supplier.SUP_NAME);
            preparedStatement.setString(3, supplier.STREET);
            preparedStatement.setString(4, supplier.CITY);
            preparedStatement.setString(5, supplier.STATE);
            preparedStatement.setString(6, supplier.ZIP);

            System.out.println(sql);

            // Don't supply sql
            preparedStatement.executeUpdate();
            DBConnections.getWarningsFromStatement(preparedStatement);
        }
        connection.commit();
        connection.setAutoCommit(true);

        System.out.printf("PreparedStatement (paramaterized) update insertion into: %s%n",
tableNameQualified);


    } catch (SQLException e) {
        if (connection != null) {
            try {
                connection.rollback();
                System.err.println("Rolledback update attempt.");
            } catch (SQLException rolle) {
                DBConnections.handleSQLException(rolle);
            }
        }
        // Duplicate entry 'X' for key 'Y'
        if (e.getSQLState().equalsIgnoreCase("23000")) {
            System.err.println(e.getMessage());
        } else {
            DBConnections.handleSQLException(e);
        }
    }
}
```

C:\Users\John\Documents\Sda\Code\Java\Examples\TutorialAtOracle\JDBCTutorial\src\jlb\DataManipulationOperations.java
(Oracle, 2017), "Using Transactions", https://docs.oracle.com/javase/tutorial/jdbc/basics/transactions.html

By default each SQL statement will auto-commit, so you need to turn off auto-commit when performing transactions and turn it on again after a commit (see above).

(Oracle, 2017), "Using Transactions", https://docs.oracle.com/javase/tutorial/jdbc/basics/transactions.html

You can set "save points" and do rollbacks to those save points.

(Oracle, 2017), " Setting and Rolling Back to Savepoints",
https://docs.oracle.com/javase/tutorial/jdbc/basics/transactions.html#set_roll_back_savepoints

RowSets

Overview

RowSets are derived from ResultSets and have the following advantages:

1. They add an updateable and scrollably capability even if the underlying RDMS does not support scrollable and updateable ResultSets.

2. They have "a set of properties and a listener notification mechanism that make it a JavaBeans component".
3. They can, depending on the type, be disconnected (in addition to being connected).

(Oracle, 2017), "Using RowSet Objects", https://docs.oracle.com/javase/tutorial/jdbc/basics/rowset.html
(Oracle, 2017), Using JdbcRowSet Objects", https://docs.oracle.com/javase/tutorial/jdbc/basics/jdbcrowset.html

The RowSet object hierarchy is as follows:

- ResultSet
  - RowSet
    - JdbcRowSet (Connected)
    - CachedRowSet (Disconnected)
      - WebRowSet (+ can read/write in XML form)
        - JoinRowSet (+ form a SQL JOIN equivalent).
        - FilteredRowSet (+ filter without execting a query).

(Oracle, 2017), "Hierarchy For Package javax.sql.rowset", https://docs.oracle.com/javase/8/docs/api/javax/sql/rowset/package-tree.html

> Disconnected RowSet objects are also serializable, and the combination of being both serializable and lightweight makes them ideal for sending data over a network. They can even be used for sending data to thin clients such as PDAs and mobile phones.

(Oracle, 2017), "Using RowSet Objects", " Disconnected RowSet Objects", https://docs.oracle.com/javase/tutorial/jdbc/basics/rowset.html

JdbcRowSet

Intro

> One of the main uses of a JdbcRowSet object is to make a ResultSet object scrollable and updatable when it does not otherwise have those capabilities.

(Oracle, 2017), Using JdbcRowSet Objects, https://docs.oracle.com/javase/tutorial/jdbc/basics/jdbcrowset.html

JdbcRowSet Creation

Concurrency

Introduction

Concurrent programming is facilitating an application to attend to more than one task at the same time. In concurrent programing there are two basic units of execution: processes and threads. Java mostly concerns itself with threads.

(Oracle, 2017) *http://docs.oracle.com/javase/tutorial/essential/concurrency/procthread.html*

All Java applications start with at least one thread, the main thread. For concurrent programming from the main thread you create further threads.

Threads

Define and start a thread

There are two basic strategies for using Thread objects to create concurrent applications: directly handle thread creation by instantiating a Thread object; abstract the thread handling by passing an application task to an *executor*.

(Oracle, 2017) *http://docs.oracle.com/javase/tutorial/essential/concurrency/threads.html*

There are two ways to define and start a thread:

1. By subclassing Thread; and

```
public class HelloThread extends Thread {

    public void run() {
        System.out.println("Hello from a thread!");
    }

    public static void main(String args[]) {
        (new HelloThread()).start();
    }

}
```

2. By implementing a Runnable object. In general use this second technique.

```
// Define a task to be used by a thread by implementing a Runnable object
public class HelloRunnable implements Runnable {
  @Override
  public void run() {
    // tasks be to executed by a thread
    System.out.println("Hello from a thread.");
  }
}

// Start the thread (from a different class).
public static void main(String[] args) {
  Thread t = new Thread(new HelloRunnable());
  t.start();
}
```

(Oracle, 2017) *http://docs.oracle.com/javase/tutorial/essential/concurrency/runthread.html*

Generally you'll want to implement a `runnable` rather than extend a `thread`, because it is more general. More general in that "the `Runnable` oject can subclass other than `Thread`".

(Oracle, 2017) *http://docs.oracle.com/javase/tutorial/essential/concurrency/runthread.html*

Pause (Sleep)

To pause execution use Thread.Sleep(). The sleep()method can be interrupted by another thread therefore you must catch or specify InterruptedException.

```
public class SleepMessages {
  public static void begin() throws InterruptedException {
    String messages[] =
        { "How now brown cow",
          "The bull by the horns",
          "The fonze is cool",
          "Now is the time for all men" };

    for (int i = 0; i < messages.length; i++) {
      System.out.println(messages[i]);
      // Pause for 4 seconds
      Thread.sleep(4000);
    }
  } // Begin
} // SleepMessages
```

(Oracle, 2017) *http://docs.oracle.com/javase/tutorial/essential/concurrency/sleep.html*

Interrupts

An interrupt is a signal to a thread and do something else. It is up to the programmer to decide what the thread should do in response to an interrupt. Frequently the thing to do is to terminate the thread.

(Oracle, 2017) *http://docs.oracle.com/javase/tutorial/essential/concurrency/interrupt.html*

Two patterns used to support interrupts....

```
// Catch an InterruptedException (for methods that throw InterruptedException)
for (int i = 0; i < importantInfo.length; i++) {
    // Pause for 4 seconds
    try {
        Thread.sleep(4000);
    } catch (InterruptedException e) {
        // We've been interrupted: no more messages.
        return;
    }
    // Print a message
    System.out.println(importantInfo[i]);
}

// Poll Thread.interrupted() (for methods that do not throw InterruptedException)
for (int i = 0; i < inputs.length; i++) {
    heavyCrunch(inputs[i]);
    if (Thread.interrupted()) {
        // We've been interrupted: no more crunching.
        return;
    }
}
```

(Oracle, 2017) *http://docs.oracle.com/javase/tutorial/essential/concurrency/interrupt.html*

When using Thread.interrupted() you will frequently want to throw a InterruptedException to centralize exception handling.

```
if (Thread.interrupted()) {
    throw new InterruptedException();
}
```

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/essential/concurrency/interrupt.html

Effect an interrupt by setting an interrupt flag with Thread.interrupt(), a non-static method. (or just throw an InterruptedException ??).

Interrupt test	**Does interrupt test reset interrupt flag**
Thread.interrupted()	Yes.
isInterrupted()	No
InterruptedException [The method that throws this clears the interrupt]	Yes. [However another thread could always set the interrupt thread immediately again]

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/essential/concurrency/interrupt.html

Thread methods that respond to an interrupt by exiting with an InterruptedException.

- sleep()
- join()

Join

Pause the current thread until another thread finishes

```
// In the current thread
t.join();

// Where t is a thread that is also executing. The current thread waits for t to finish
before moving on.
```

(Oracle, 2017) http://docs.oracle.com/javase/tutorial/essential/concurrency/join.html

Thread Example

Thread Example code.

```
public class SimpleThreads {

  private static void outputThreadMessage(String message){
    String threadName = Thread.currentThread().getName();
    System.out.format("%8s: %s%n", threadName, message);
  }

  private static class MessageLoop implements Runnable {
    @Override
    public void run() {
      String messages[] =
        { "How now brown cow",
          "The bull by the horns",
          "The fonze is cool",
          "Now is the time for all men" };

      try {
        for (int i = 0; i < messages.length; i++) {
          // Pause for 4 seconds
          Thread.sleep(4000);
```

```
          outputThreadMessage(messages[i]);
        }
      } catch (InterruptedException e) {
          outputThreadMessage("I wasn't done yet");
      } // try-catch

    } // run
  } // MessageLoop

  public static void begin(String[] args) throws InterruptedException {
    // Delay, in milliseconds before we interrupt the MessageLoop thread
    // (default one hour)
    long patience = 1000 * 60 * 60;

    outputThreadMessage("Starting MessageLoop");
    long startTime = System.currentTimeMillis();
    Thread t = new Thread(new MessageLoop());
    t.start();

    // If command line argument present, gives patience in seconds
    if (args.length > 0) {
      try {
        patience = Long.parseLong(args[0]) * 1000;
      } catch (NumberFormatException e) {
        System.err.println("Argument must be an integer");
        System.exit(1);
      }
    }

    while (t.isAlive()) {
      outputThreadMessage("Still waiting ...");
      // Wait for 1 second for MessageLoop to finish
      t.join(1000);

      if (((System.currentTimeMillis() - startTime) > patience) && t.isAlive()) {
        outputThreadMessage("Tired of waiting!");
        t.interrupt();
        // Wait indefinitely
        t.join();
      }
    } // While
    outputThreadMessage("Finally done.");
  } // begin()
}
```

Thread Example output

```
C:\Users\...\Concurrency\bin>java au.com.softmake.concurrency.Concurrency 9
    main: Starting MessageLoop
    main: Still waiting ...
    main: Still waiting ...
    main: Still waiting ...
    main: Still waiting ...
Thread-0: How now brown cow
    main: Still waiting ...
    main: Still waiting ...
    main: Still waiting ...
    main: Still waiting ...
Thread-0: The bull by the horns
    main: Still waiting ...
    main: Tired of waiting!
Thread-0: I wasn't done yet
    main: Finally done.
```

References (Microsoft)

Bentley, J. (2004, Aug 12). EcmaScript in Web Quick Reference.

Bentley, J. (2013, Jul 05). Java Reference - Language - Printf Style Formatting.docx. Retrieved from "C:\Users\John\Documents\Sda\Info\Java\KB\Reference\Java Reference - Printf Style Formatting.docx"

Bentley, J. (2013, Jul). Java Reference - Language - Regular Expressions.docx.

Bentley, J. (2013, Jul). TutorialAtOracle Code Examples. Retrieved from C:\Users\John\Documents\Sda\Code\Java\Examples\TutorialAtOracle\

Bentley, J. (2013, Jul). TutorialAtOracle Code Examples - PasswordConsole. Retrieved from C:\Users\John\Documents\Sda\Code\Java\Examples\TutorialAtOracle\PasswordConsole

Bentley, J. (2016, Apr 06). Java Reference - Java EE.

Bentley, J. (2017, Dec 29). Java Reference - Language.docx.

Goyvaert, J. (2009, Jun 17). Retrieved Jul 10, 2013, from regular-expressions.info: http://www.regular-expressions.info/

joda.org. (2013, Aug). *Joda User Guide*. Retrieved Aug 14, 2013, from joda.org: http://joda-time.sourceforge.net/userguide.html

Oracle. (2011). *Java Platform, Standard Edition 6, API Specification*. Retrieved Jul 09, 2013, from http://docs.oracle.com: http://docs.oracle.com/javase/6/docs/api/overview-summary.html

Oracle. (2014). Retrieved from Java Platform, Enterprise Edition: The Java EE Tutorial: https://docs.oracle.com/javaee/7/tutorial/index.html

Oracle. (2014). Retrieved from Java Platform, Enterprise Edition: Your First Cup: An Introduction to the Java EE Platform: https://docs.oracle.com/javaee/7/firstcup/index.html

Oracle. (2017). *Java Platform, Standard Edition 8, API Specification.* Retrieved from http://docs.oracle.com: http://docs.oracle.com/javase/7/docs/api/overview-summary.html

Oracle. (2017). *The Java Tutorials.* Retrieved Jun 14, 2012, from http://docs.oracle.com/: http://docs.oracle.com/javase/tutorial/

Orcale. (2007). Retrieved 01 13, 2015, from Oracle Database XML Java API Reference: http://docs.oracle.com/cd/B28359_01/appdev.111/b28391/toc.htm

Schreckmann, D. (2003, Mar 31). *An Introduction to java.util.regex, Part 2: More Pattern Elements*. Retrieved Jul 10, 2013, from Java Ranch: http://www.javaranch.com/journal/2003/04/RegexTutorial.htm

Stackoverflow. (2018). Retrieved from Stackoverflow: https://stackoverflow.com/

W3C. (2000, Nov 12). Retrieved Jan 01, 2015, from Document Object Model (DOM) Level 2 Traversal and Range Specification, Version 1.0, W3C Recommendation 13 November, 2000: http://www.w3.org/TR/2000/REC-DOM-Level-2-Traversal-Range-20001113/

Document Licence

Java Reference - Framework © 2021 by John Bentley is licensed under Attribution-NonCommercial-ShareAlike 4.0 International