C.I.F.P A Carballeira
Acceso a datos
Práctica 2

Trabajar con ficheros JSON

Profesora:
Sandra Pereira Álvarez

Alumno:
Toro Moros, Kevin Oliver

Índice

Definición del contexto de la aplicación

La aplicación recoge la siguiente serie de información meteorológica, de 7 ciudades, A coruña, Lugo, Ourense, Pontevedra, Vigo, Santiago de Compostela y Ferrol de un JSON proporcionado por meteogalicia, teniendo esto claro, los datos que imprime serían los siguientes:

- Nombre de la ciudad.
- Id del Concello.
- Estado del cielo.
- Temperatura máxima.
- Temperatura mínima.
- Viento.
- Precipitaciones.
- Fecha de predicción.

Librerías JSON y CSV

En mi clase llamada PeticiónApi utilice las siguientes librerías:

```
// Importes necesarios para desarrollar la clase
import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStreamReader;
import java.net.HttpURLConnection;
import java.net.URL;
import org.json.JSONException;
import org.json.JSONObject;
```

- **Java.io:**
  - BufferredReader: Clase que nos proporciona un método para leer caracteres de un flujo de entrada de caracteres.
  - IOException: Clase de excepción que se lanza cuando ocurre algún error durante las operaciones de entrada y salida.
  - InputStreamReader: Clase que facilita la conversión de bytes a caracteres.
- **java.net:**
  - HttpURLConnection: Clase que proporciona una interfaz para poder realizar solicitudes HTTP a un servidor y también para leer la respuesta de dicho servidor.
  - URL: Clase para poder manipular urls creando un objeto de tipo URL.
- **org.json:**
  - JSONException: Clase que para manejar las excepciones, es decir, los errores que pueden haber al manipular datos JSON.

- JSONObject: Esta clase proporciona métodos para manipular y acceder a los datos en formato JSON.

En mi clase DatosClimaJson utilicé las siguientes librerías:

```
import java.io.IOException;
import java.time.LocalDate;
import java.time.format.DateTimeFormatter;
import java.util.ArrayList;
import java.util.List;
import org.json.JSONArray;
import org.json.JSONException;
import org.json.JSONObject;
```

- **java.time**
  - LocalDate: Clase que proporciona métodos para manipular fechas, por ejemplo, extraer el día, mes y año actual.
  - format.DateTimeFormatter: Clase que me permite formatear datos de tipo date.
- **java.util**
  - Arraylist: Clase que me proporciona métodos para agregar, eliminar, acceder y manipular elementos de una lista.
  - List: Clase que proporciona métodos para trabajar con listas, como add, get o remove.
- **org.json**
  - JSONArray: Clase que proporciona métodos para acceder y manipular los elementos de un array, así como construir y parsear arrays JSON.

En mi clase EscrituraCSV utilice las siguientes librerias:

```
import java.io.FileWriter;
import java.util.List;
import com.opencsv.CSVWriter;
```

- **java.io**
  - FileWriter: Clase que me permite escribir en un archivo.
- **com.opencsv**
  - CSVWriter: Clase que me permite escribir datos en formato CSV.

Uso API y funcionalidad

La API de meteogalicia posee una url para cada concello que en nuestro caso serían 7 url, extraje información meteorológica de esos 7 concellos, la pedida en el ejercicio y otras a mayores, porque no había información sobre humedad relativa ni sobre la cobertura nubosa.

Una vez extraídos los datos de la api se muestran por pantalla y posteriormente se genera un documento CSV con dichos datos.

Proceso de desarrollo

Primero comencé creando el proyecto en maven para poder agregarle las dependencias necesarias para poder trabajar con el JSON y para poder trabajar con archivos CSV que son las siguientes:

```
<!-- https://mvnrepository.com/artifact/org.json/json -->
<dependencies>
    <dependency>
        <groupId>org.json</groupId>
        <artifactId>json</artifactId>
        <version>20230227</version>
    </dependency>

    <!-- Dependencia CSV -->
    <dependency>
        <groupId>com.opencsv</groupId>
        <artifactId>opencsv</artifactId>
        <version>5.5</version>
    </dependency>
</dependencies>
```

Luego de agregar las dependencias necesarias, cree mi clase Clima que tendría las propiedades que quiero que se extraigan del JSON de meteogalicia

```java
public class Clima implements Serializable {

    // Creo una variable de tipo String llamada nome
    private String nome;
    // Creo una variable de tipo int llamada idConcello
    private int idConcello;
    // Creo una variable de tipo int llamada estadoCeo
    private int estadoCeo;
    // Creo una variable de tipo double llamada temperaturaMax
    private double temperaturaMax;
    // Creo una variable de tipo double llamada temperaturaMin
    private double temperaturaMin;
    // Creo una variable de tipo double llamada vento
    private double vento;
    // Creo una variable de tipo double llamada precipitacions
    private double precipitacions;
    // Creo una variable de tipo string llamada fecha
    private String fecha;

    // Creo un constructor para la clase con sus respectivos parametros,
    // inicializando de esta manera las variables al crear una instancia de Clima
    public Clima(String nome, int idConcello, int estadoCeo, double temperaturaMax, double temperaturaMin, double vento,
            double precipitacions, String fecha) {
        this.nome = nome;
        this.idConcello = idConcello;
        this.estadoCeo = estadoCeo;
        this.temperaturaMax = temperaturaMax;
        this.temperaturaMin = temperaturaMin;
        this.vento = vento;
        this.precipitacions = precipitacions;
        this.fecha = fecha;
    }
    // Creo los getters y setters de la clase Clima
```

Una vez creada mi clase clima creé una clase PeticiónAPI, que en dicha clase hago un método de tipo JSONObject para hacer una una solicitud a la las url que le pasarían por parámetro, esta solicitud la hago mediante el setRequestMethod("GET").

Luego creo un BufferedReader para leer la respuesta de la solicitud y luego convertirla a caracteres legibles con el getInputStream.

Hago un bucle while para leer línea por línea hasta que no hayan más y se agrega al StringBuilder y cierro el BufferedReader después de haber leído toda la información

Creo un objeto de tipo JSONObject que contendrá la respuesta de la url y al final me vuelve un JSONObject, que tiene la respuesta de url en formato JSON.

```java
public class PeticionApi {

    // Método para realizar una solicitud HTTP GET y devolver la respuesta como objeto JSONObject
    public static JSONObject DatosAPI(String url) throws IOException, JSONException {

        // Creación de una URL que por parametro se le pasa la String url
        URL urls = new URL(url);
        // Apertura de la conexión HTTP a la URL
        HttpURLConnection conn = (HttpURLConnection) urls.openConnection();
        // Realización de la solicitud para obetener información de la url
        conn.setRequestMethod(method:"GET");

        // Creación de un BufferedReader para leer la respuesta de la url, luego convertir esa respuesta en caracteres legibles
        BufferedReader rea = new BufferedReader(new InputStreamReader(conn.getInputStream()));
        // Creación de un StringBuilder para guardar la respuesta de la url
        StringBuilder res = new StringBuilder();

        String linea;
        // Leemos línea por línea hasta que no hayan mas lineas por leer
        while ((linea = rea.readLine()) != null) {
            //se agrega al StringBuilder res
            res.append(linea);
        }
        // Se cierra el BufferedReader después de haber leido toda la información
        rea.close();

        // Creación de un objeto de tipo JSONObject, que contiene la respuesta de la url como un String
        JSONObject jsonObject = new JSONObject(res.toString());

        // Devuelve el objeto JSONObject, que tiene la respuesta de la url en formato JSON
        return jsonObject;

    }
}
```

Luego creo una clase llamada DatosClima que tiene un método que convierte un objeto JSON a uno de tipo Clima, que se llama objetoDatosjson y se le pasa por parámetro el objeto JSON creado en la clase anterior. El método crea y devuelve un objeto de tipo Clima con la información que se obtuvo del JSON de la API.

```java
public class DatosClimaJson {

    // Método para convertir un JSONObject en un objeto de tipo Clima
    public static Clima objetoDatosjson(JSONObject json) {

        // Obtenemos la lista de predicciones diarias para el concello
        JSONArray predDiaConcelloList = json.getJSONObject(key:"predConcello").getJSONArray(key:"listaPredDiaConcello");
        // Obtenemos la primera prediccion del concello
        JSONObject pred = predDiaConcelloList.getJSONObject(index:0);
        // Obtenemos el nombre de la ciudad
        String nome = json.getJSONObject(key:"predConcello").getString(key:"nome");
        // Obtenemos el id del Concello
        int idConcello = json.getJSONObject(key:"predConcello").getInt(key:"idConcello");
        // Obtenemos el estado del cielo en este caso en la mañana
        int estadoCeo = pred.getJSONObject(key:"ceo").getInt(key:"manha");
        // Obtenemos la temperatura máxima
        int temperaturaMax = pred.getInt(key:"tMax");
        // Obtenemos la temperatura minima
        int temperaturaMin = pred.getInt(key:"tMin");
        // Obtenemos el viento en este caso en la mañana
        int vento = pred.getJSONObject(key:"vento").getInt(key:"manha");
        // Obtenemos la precipitacion en este caso en la mañana
        int precipitacions = pred.getJSONObject(key:"pchoiva").getInt(key:"manha");
        // Obtenemos la fecha de la prediccion
        String fecha = pred.getString(key:"dataPredicion");
        // Convierto la el String fecha a un objeto LocalDate
        LocalDate fechaPred = LocalDate.parse(fecha, DateTimeFormatter.ISO_DATE_TIME);
        // Formateo la fecha
        String fechaFormateada = fechaPred.format(DateTimeFormatter.ofPattern(pattern:"dd/MM/yyyy"));
        // Creo y devuelvo un objeto de tipo Clima con la información anteriormente obtenida
        return new Clima(nome, idConcello, estadoCeo, temperaturaMax, temperaturaMin, vento, precipitacions, fechaFormateada);

    }
```

En la misma clase creo un método para obtener una lista de objetos de tipo Clima a partir de una lista de urls. Me devuelve una lista de objetos de tipo clima.

```java
//Método para obtener una lista de objetos de tipo Clima a partir de una lista de urls
public static List<Clima> climaLista(List<String> urls) throws JSONException, IOException {

    // Creación de una lista para almacenar objetos de tipo Clima
    List<Clima> clima = new ArrayList<>();
    // Foreach para iterar cada url en la lista
    for (String url : urls) {
        // Solicitud HTTP a la url y obtenemos el objeto JSON como respuesta
        JSONObject json = PeticionApi.DatosAPI(url);
        // Convierto el objeto JSON a un objeto Clima y lo agrego a la lista
        clima.add(objetoDatosjson(json));
    }
    // Devuelvo la lista de objetos de tipo Clima
    return clima;
}
```

Luego creo un método que muestre la información de una lista de objetos de tipo clima, en el cual creo un foreach para que recorra toda la lista y me los muestre utilizando el método mostrarInformaciónClima.

```java
// Método para mostrar la información de una lista de objetos Clima
public static void mostrarInfoClima(List<Clima> climas) {
    // Iterar cada objeto clima en la lista
    for (Clima clima : climas) {
        // Llamo al metodo mostrarInformacionClima, imprimo la información del objeto Clima
        mostrarInformacionClima(clima);
        System.out.println(x:"");
    }
}
```

Creo el método mostrarInformación el cual tendrá como parametro un objeto de tipo clima e imprimirá el toString de la clase Clima.

```java
// Método para mostrar la información de un objeto Clima
public static void mostrarInformacionClima(Clima clima) {
    // Imprimo el toString del objeto CLima
    System.out.println(clima.toString());
}
```

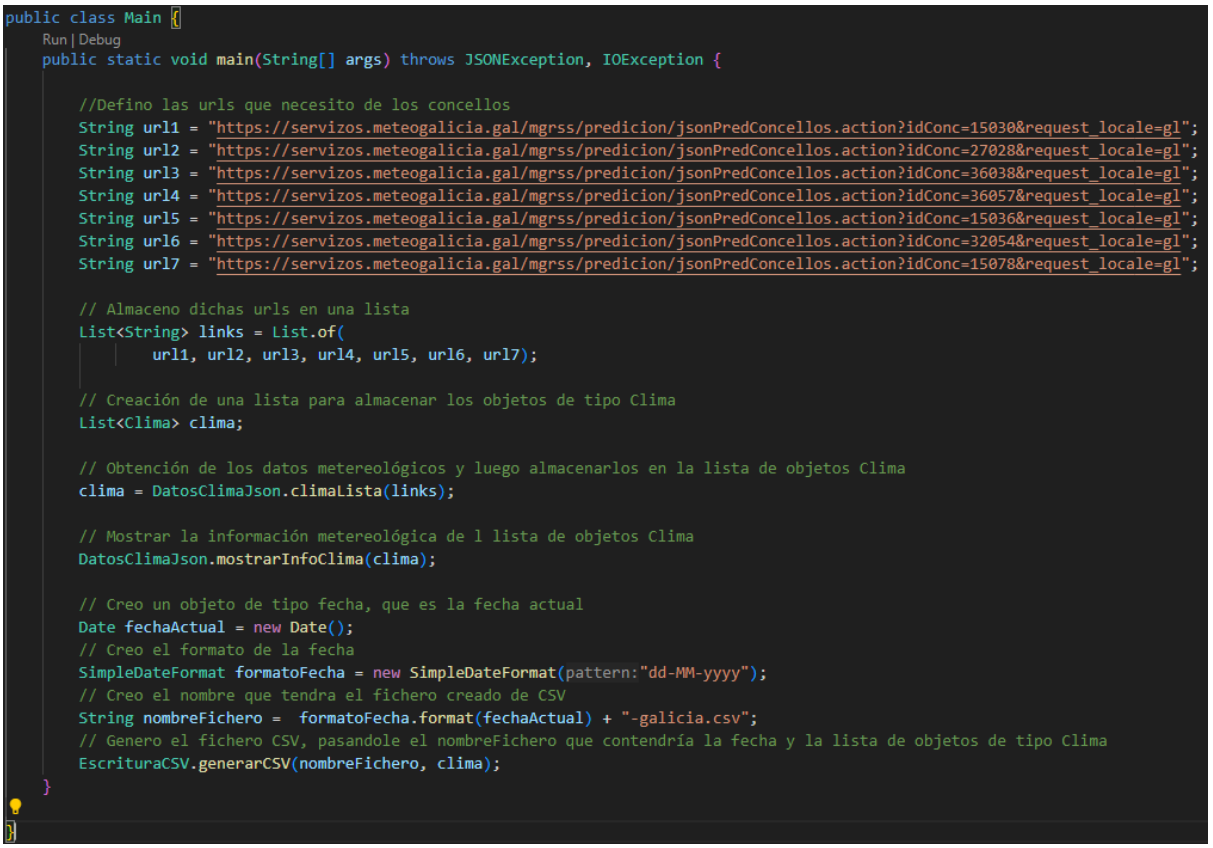
Por último creo la clase EscrituraCSV, la cual tiene el método generarCSV que lo que hace es crear un fichero de tipo CSV con un encabezado que contiene la información que se ve en la imagen, luego recorro la lista de objetos clima y escribo la información en el archivo CSV.

```java
public class EscrituraCSV {
    // Método para generar el fichero CSV, pasandole como parametro un nombre y una lista de objetos Clima
    public static void generarCSV(String nombreFichero, List <Clima> climaCon){

        // Apertura para escribir en un fichero CSV
        try (CSVWriter escritura = new CSVWriter(new FileWriter(nombreFichero))) {
            // Defino el encabezado del archivo CSV
            String[] header = {"Ciudad", "Id del Concello", "Estado do ceo","Temperatura Maxima","Temperatura Minima","Vento","Precipitacions","Fecha de la prediccion"};
            // Escribo el encabezado en el archivo CSV
            escritura.writeNext(header);
            // Foreach que Itera sobre la lista de objetos Clima y escribe la información en el archivo CSV
            for (Clima climaConcello : climaCon){
                String[] info = {
                    String.valueOf(climaConcello.getNome()),
                    String.valueOf(climaConcello.getIdConcello()),
                    String.valueOf(climaConcello.getEstadoCeo()),
                    String.valueOf(climaConcello.getTemperaturaMax()),
                    String.valueOf(climaConcello.getTemperaturaMin()),
                    String.valueOf(climaConcello.getVento()),
                    String.valueOf(climaConcello.getPrecipitacions()),
                    String.valueOf(climaConcello.getFecha()),
                };
                // Escribo la información del objeto Clima en una nueva línea del archivo CSV
                escritura.writeNext(info);
            }
            // Imprimo un mensaje indicando que el fichero CSV se ha creado correctamente
            System.out.println(nombreFichero + ". Fichero CSV creado correctamente.");
        } catch (Exception e) {
            // TODO: handle exception
            e.printStackTrace();
        }
    }
}
```

Mi clase main sería la que se aprecia en la imagen, creando primero las variables de las url que necesito para sacar la información de ellas. Posteriormente creo una lista en donde almaceno las siete url y luego creo una lista para almacenar los datos de tipo Clima, con la lista creada de objetos Clima, obtengo los datos meteorológicos y los almaceno en dicha lista y muestro la información meteorológica de la lista de objetos Clima.

Por último creo un objeto de tipo Date para obtener la fecha actual, la cual le coloco un formato adecuado y luego creo el nombre que tendrá el fichero csv.



```java
public class Main {
    Run | Debug
    public static void main(String[] args) throws JSONException, IOException {

        //Defino las urls que necesito de los concellos
        String url1 = "https://servizos.meteogalicia.gal/mgrss/predicion/jsonPredConcellos.action?idConc=15030&request_locale=gl";
        String url2 = "https://servizos.meteogalicia.gal/mgrss/predicion/jsonPredConcellos.action?idConc=27028&request_locale=gl";
        String url3 = "https://servizos.meteogalicia.gal/mgrss/predicion/jsonPredConcellos.action?idConc=36038&request_locale=gl";
        String url4 = "https://servizos.meteogalicia.gal/mgrss/predicion/jsonPredConcellos.action?idConc=36057&request_locale=gl";
        String url5 = "https://servizos.meteogalicia.gal/mgrss/predicion/jsonPredConcellos.action?idConc=15036&request_locale=gl";
        String url6 = "https://servizos.meteogalicia.gal/mgrss/predicion/jsonPredConcellos.action?idConc=32054&request_locale=gl";
        String url7 = "https://servizos.meteogalicia.gal/mgrss/predicion/jsonPredConcellos.action?idConc=15078&request_locale=gl";

        // Almaceno dichas urls en una lista
        List<String> links = List.of(
                url1, url2, url3, url4, url5, url6, url7);

        // Creación de una lista para almacenar los objetos de tipo Clima
        List<Clima> clima;

        // Obtención de los datos metereológicos y luego almacenarlos en la lista de objetos Clima
        clima = DatosClimaJson.climaLista(links);

        // Mostrar la información metereológica de l lista de objetos Clima
        DatosClimaJson.mostrarInfoClima(clima);

        // Creo un objeto de tipo fecha, que es la fecha actual
        Date fechaActual = new Date();
        // Creo el formato de la fecha
        SimpleDateFormat formatoFecha = new SimpleDateFormat(pattern:"dd-MM-yyyy");
        // Creo el nombre que tendra el fichero creado de CSV
        String nombreFichero =  formatoFecha.format(fechaActual) + "-galicia.csv";
        // Genero el fichero CSV, pasandole el nombreFichero que contendría la fecha y la lista de objetos de tipo Clima
        EscrituraCSV.generarCSV(nombreFichero, clima);
    }
}
```

Prueba

Este sería el resultado por pantalla:

```
        Clima
Ciudad: A Coruña
Id del Concello: 15030
Estado do ceo: 104
Temperatura Maxima: 14.0
Temperatura Minima: 7.0
Vento: 304.0
Precipitacions: 5.0
Fecha de la prediccion: 19/12/2023

        Clima
Ciudad: Lugo
Id del Concello: 27028
Estado do ceo: 115
Temperatura Maxima: 8.0
Temperatura Minima: -1.0
Vento: 299.0
Precipitacions: 5.0
Fecha de la prediccion: 19/12/2023

        Clima
Ciudad: Pontevedra
Id del Concello: 36038
Estado do ceo: 103
Temperatura Maxima: 14.0
Temperatura Minima: 3.0
Vento: 299.0
Precipitacions: 5.0
Fecha de la prediccion: 19/12/2023
```

```
        Clima
Ciudad: Vigo
Id del Concello: 36057
Estado do ceo: 103
Temperatura Maxima: 13.0
Temperatura Minima: 4.0
Vento: 299.0
Precipitacions: 5.0
Fecha de la prediccion: 19/12/2023

        Clima
Ciudad: Ferrol
Id del Concello: 15036
Estado do ceo: 103
Temperatura Maxima: 13.0
Temperatura Minima: 6.0
Vento: 306.0
Precipitacions: 5.0
Fecha de la prediccion: 19/12/2023

        Clima
Ciudad: Ourense
Id del Concello: 32054
Estado do ceo: 106
Temperatura Maxima: 10.0
Temperatura Minima: -1.0
Vento: 299.0
Precipitacions: 5.0
Fecha de la prediccion: 19/12/2023

        Clima
Ciudad: Santiago de Compostela
Id del Concello: 15078
Estado do ceo: 103
Temperatura Maxima: 11.0
Temperatura Minima: 2.0
Vento: 300.0
Precipitacions: 5.0
Fecha de la prediccion: 19/12/2023
```

Y el resultado del CSV el siguiente:

19-12-2023-galicia.csv

```
"Ciudad","Id del Concello","Estado do ceo","Temperatura Maxima","Temperatura Minima","Vento","Precipitacions","Fecha de la prediccion"
"A Coruña","15030","104","14.0","7.0","304.0","5.0","19/12/2023"
"Lugo","27028","115","8.0","-1.0","299.0","5.0","19/12/2023"
"Pontevedra","36038","103","14.0","3.0","299.0","5.0","19/12/2023"
"Vigo","36057","103","13.0","4.0","299.0","5.0","19/12/2023"
"Ferrol","15036","103","13.0","6.0","306.0","5.0","19/12/2023"
"Ourense","32054","106","10.0","-1.0","299.0","5.0","19/12/2023"
"Santiago de Compostela","15078","103","11.0","2.0","300.0","5.0","19/12/2023"

```