

# Proyecto01: Web Service

Integrantes:

Arizmendi López Alexis de Jesús

Cortés Jiménez Carlos Daniel

Cruz Hernández Víctor Emiliano

10/03/2022.

## 1. Definición del problema

En este proyecto nos centramos en la elaboración de un web service para el aeropuerto de la ciudad de México, en el cual se realizará un programa para consultar características, e información relevante de un punto A a otro punto B (**en este caso ciudades**) para ello nos apoyaremos sobre otro web service para conocer la información requerida para la solución del problema planteado, como lo es OpenWeatherMap que nos proporciona datos meteorológicos históricos y actuales.

## 2. Análisis del problema

Empezamos por el informe del clima de la ciudad, para lograr esto como anteriormente se mencionó utilizaremos OpenWeatherMap para mostrar informe de la ciudad en la que se encuentra el cliente o usuario y desplegarla por pantalla características como temperatura de la ciudad, la humedad, la presión, la longitud, latitud y el tipo de clima en la ciudad que se encuentra así como también mostrar todas estas características a la ciudad a la que se dirige dicha persona, un dato importante a resaltar es que el programa no es interactivo, esto quiere decir que, el usuario no podrá manipular dicha aplicación, si no que la aplicación misma desplegará toda la información sin ser manipulada.

Además este despliegue de información deberá realizarse para 3000 ticks una vez que corra el programa, también deberemos contar con un archivo .csv donde se tenga el código IATA de las ciudades a las que se puede realizar un viaje, para esto se hará dos clases una que contenga la estructura para obtener el clima y otra para los vuelos (que contiene el código IATA, longitud y latitud de la ciudad origen y destino).

Otra dato a resaltar es que se deberá realizar un cache para guardar todos aquellos datos que sean recurrentes para así acceder de forma más rápida a ellos. Una vez se tengan hecho todo lo anterior dicho se mostrará en terminal por medio de una tabla toda la información que deberá contener nuestro ticket

## 3. Selección de la mejor alternativa

Para la resolución de este proyecto decidimos usar Apache Maven ya que es un software enfocado a la gestión y manipulación para la creación de proyectos en Java, así como para facilitarnos el uso de librerías de terceros, y además ayuda mucho a simplificar la creación de código, y además nos ayuda a validar, compilar, "testear" (Pruebas unitarias), verificar y empaquetar nuestro proyecto, dados estos puntos es por lo que nos decidimos por usarlo para la resolución rápida y eficaz del proyecto.

## 4. Pseudocódigo

1.- Inicio

```
2.- rutaCSV = "src/main/resources/archivoBoletos/dataset1.csv";
3.- maximoNumeroPeticiones = 55;
4.- maxTiempoCache = ( 1000 * 60 * 60 * 5 );
5.- diccionarioRegiones = new HashMap<String, Lugar>();
6.- listaVuelos = new HashMap<String, Vuelo>();
7.- diccionarioClimas = new HashMap<String, Clima>();

8.- meterADiccionarios(String sOrigen, String sDestino, String so_lat, String so_lon, String sd_lat, String sd_lon);
```

```
Inicializamos Lugar origen = null;
Inicializamos Lugar destino = null;
```

8.1.- Si diccionarioRegiones.containsKey(sOrigen)

```
origen = diccionarioRegiones.get(sOrigen);
```

De lo contrario:

```
origen = new Lugar(sOrigen, so_lat, so_lon);
diccionarioRegiones.put(sOrigen, origen);
```

```
Clima clima = new Clima(sOrigen, so_lat, so_lon);
diccionarioClimas.put(sOrigen, clima);
```

8.1.1.- Si diccionarioRegiones.containsKey(sDestino)

```
destino = diccionarioRegiones.get(sDestino);
```

De lo contrario haz:

```
destino = new Lugar(sOrigen, sd_lat, sd_lon);
diccionarioRegiones.put(sDestino, destino);
Clima clima = new Clima(sDestino, sd_lat, sd_lon);
diccionarioClimas.put(sDestino, clima);
```

Además

```
int id = Vuelo.getNuevoIdDeVuelo();
8.1.1.1.- Mostramos "id :"+id
Vuelo vuelo = new Vuelo(origen, destino), id;
listaVuelos.add(vuelo);
```

9.- llenarValoresDiccionarioClima()

```
9.1.- Mostramos: diccionarioClimas.size();
9.2.- Mostramos: Cargando peticiones..."
```

```
inicializamos contador = 0;
```

9.3.- Mientras ap.Entry<String,Clima> v: diccionarioClimas.entrySet() realizamos

9.3.1.- Intenta

9.3.1.1.- Si contador>0 and contador %maximoNumeroPeticiones==0

9.3.1.1.1.- Mostramos: "Hay más de "+maximoNumeroPeticiones+"  
peticiones, hay que esperar 1 min entre ellas.";

9.3.1.1.2.- Mostramos: ".<sup>Es</sup>perando para un bloque";  
Thread.sleep(1000\*60);

Ademas

v.getValue().llenarAtributos();  
contador++;

9.3.2.- Atrapa InterruptedException e

9.3.2.1.- Mostramos "ERROR en Esperar peticiones";

9.4.- Mostramos: "Terminó"

10.- imprimirMenu()

10.1.- Mostramos: "\*\*\*\*\*VUELOS\*\*\*\*\*";

10.2.- Mientras Vuelo v: listaVuelos realizamos

String origen = v.getOrigen().getId();  
String destino = v.getDestino().getId();

String temp\_origen = diccionarioClimas.get(origen).getTemperatura();  
String temp\_destino = diccionarioClimas.get(destino).getTemperatura();

String clima\_origen = diccionarioClimas.get(origen).getTipoClima();  
String clima\_destino = diccionarioClimas.get(destino).getTipoClima();

String descClima\_origen = diccionarioClimas.get(origen).getDescripcionDelClima();  
String descClima\_destino = diccionarioClimas.get(destino).getDescripcionDelClima();

String presion\_origen = diccionarioClimas.get(origen).getPresionA();  
String presion\_destino = diccionarioClimas.get(destino).getPresionA();

String humedad\_origen = diccionarioClimas.get(origen).getHumedad();  
String humedad\_destino = diccionarioClimas.get(destino).getHumedad();

dataO = String.format(  
"%-10s %-10s %-15s %-10s %-10s %-10s",  
origen, clima\_origen, temp\_origen, descClima\_origen, presion\_origen, humedad\_origen  
);

dataD = String.format(  
"%-10s %-10s %-15s %-10s %-10s %-10s",  
destino, clima\_destino, temp\_destino, descClima\_destino, presion\_destino,  
humedad\_destino  
);

10.3.- Mostramos: Vuelo de "+origen+" a "+destino+". Ticket: "+v.getId() ;

10.4.- Mostramos: "->Origen: "+dataO+"/n->Destino: "+dataD+"/n";

11.- main( String[] args )

```
Date ahora = new Date();
Date primeraEjecucionDelDia = EscritorLectorCache.leerCacheHoraEjecucionDelDia();
long tiempo24Horas = (1000 * 60 * 60 * 24);
long diferencia = 0;
```

11.1.- Si primeraEjecucionDelDia != null entonces

```
diferencia = ahora.getTime() - primeraEjecucionDelDia.getTime();
```

De lo contrario

```
diferencia = tiempo24Horas *2;
```

11.1.1.- Si diferencia >= tiempo24Horas

11.1.1.1.- Mostramos "Crea desde csv nuevo";

```
CSVReader reader = null;
```

11.1.1.1.- Intenta

```
reader = new CSVReader( new FileReader(rutaCSV.replaceAll("/", "/")) );
String[] linea = reader.readNext();
```

```
Mientras (linea = reader.readNext()) != null
```

```
sOrigen = linea[0];
sDestino = linea[1];
so_lat = linea[2];
so_lon = linea[3];
sd_lat = linea[4];
sd_lon = linea[5];
```

101.1.1.1.1.- Mostramos "Mete algo"

```
meterADiccionarios(sOrigen, sDestino, so_lat, so_lon, sd_lat, sd_lon);
```

11.1.1.2.- Atrapa FileNotFoundException e

11.1.1.2.1.- Mostramos: ".Error al leer CSV"

11.1.1.3.- Atrapa IOException ee

11.1.1.3.1.- Mostramos: "IOException"

11.1.1.4.- Atrapa CsvValidationException eee

11.1.1.4.1.- Mostramos: "CsvValidationExpection"

Además

```
llenarValoresDiccionarioClima();
```

```

EscritorLectorCache.escribirUltimaFechaEjecucion(ahora);
EscritorLectorCache.escribirFechaEjecucionDelDia(ahora);

EscritorLectorCache.escribirClimas(
    (HashMap<String,Clima>) diccionarioClimas
);
EscritorLectorCache.escribirLugares(
    (HashMap<String,Lugar>) diccionarioRegiones
);
EscritorLectorCache.escribirVuelos(
    listaVuelos

);

11.1.1.4.2.- Mostramos "número de tickets: "+listaVuelos.size();
11.1.1.4.3.- Mostramos "número de climas: "+diccionarioClimas.size();
11.1.1.4.4.- Mostramos "número de regiones: "+diccionarioRegiones.size();
imprimirMenu();

```

De lo contario haz

```

Date horaActual = new Date();
Date horaAnterior = EscritorLectorCache.leerCacheUltimaHoraEjecucion();

long diferencia2 = horaActual.getTime() - horaAnterior.getTime();

listaVuelos = EscritorLectorCache.leerCacheVuelos();
diccionarioRegiones = EscritorLectorCache.leerCacheLugares();
diccionarioClimas = EscritorLectorCache.leerCacheClima();

```

11.2.- Si diferencia2 >= maxTiempoCache entonces

11.2.1.- Mostramos "más de "+maxTiempoCache+" nueva búsqueda"

```

llenarValoresDiccionarioClima();

EscritorLectorCache.escribirUltimaFechaEjecucion(horaActual);
EscritorLectorCache.escribirClimas(
    (HashMap<String,Clima>) diccionarioClimas
);

imprimirMenu();

```

11.- Fin

## 5. Finalmente piensa a futuro

Se podrian mejorar varias muchos aspectos uno de ellos seria aumentar la capacidad de tickets que se tiene en primera instancia del problema, ya que esto podria provocar un saturamiento de la pagina y poder hacerla colapsar, otro punto seria poder hacerla interactiva algo con lo que el usuario o persona pueda sentirse mas comodo al usar nuestros servicios, implemnetar una interfaz grafica mas amigable para el usuario, tambien requerira a futuro mantenimiento para una version actualizada a dicho web service que se realizara, actualizacion de redes, implementar una pagina web.