Argentina programa 4.0

Tipos de Archivo

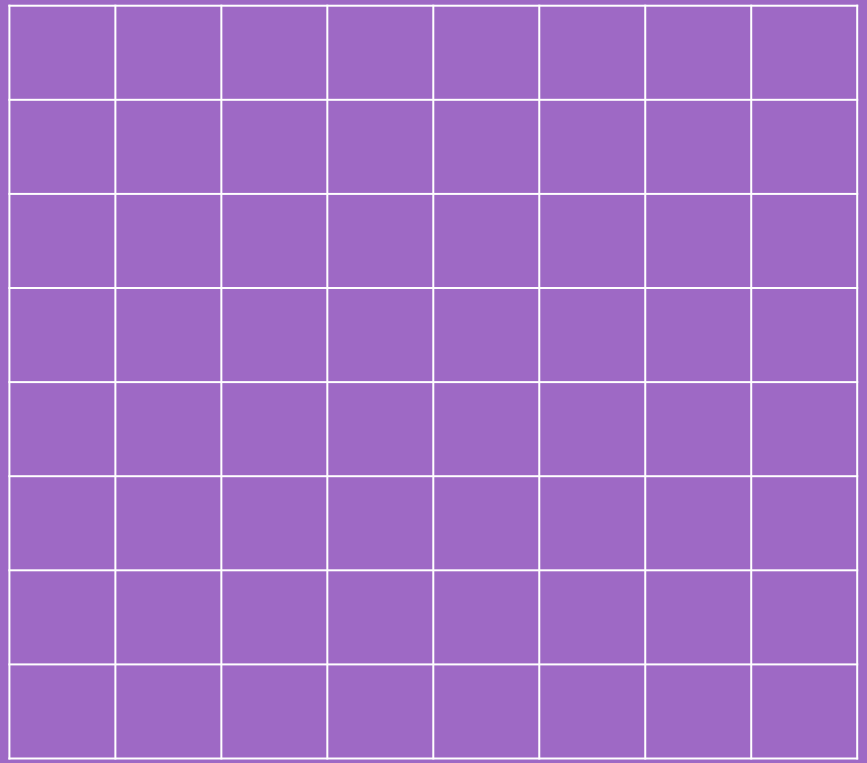
"Desarrollador Java Inicial"

Agenda

Argentina programa 4.0

- Tipos de Archivos
- Props / XML / JSON / Librerías de serialización

Archivos XML y JSON



Tipos de Archivos

Argentina programa 4.0

- En este curso trabajamos con archivos de texto, en general separados por algún tipo de delimitados
- Repasando un poco, un archivo conceptualmente, es un conjunto de datos, y dependen que el software que los procese entienda su contenido, es decir, pueda procesar su formato. Por ejemplo mostrar una imagen, procesar un texto, o ejecutar un programa Java, por ejemplo cuando la JRE ejecuta un .class
- La mayor parte de los formatos, no pueden abrirse con un procesador de texto, por ejemplo PDFs, PNGs, MP3s, ZIPs, etc, pero pueden abrirse con el programa diseñado para entenderlos

Archivos de Texto estructurados

Argentina programa 4.0

- Muchas veces se necesita transmitir o almacenar información estructurada, que sea fácil de leer y validar por lo programas, pero al mismo tiempo, “entendible” por una persona
- Existen muchas ocasiones que las “tablas”, por ejemplo los archivos csv o tsv pueden cumplir esta función, pero hay otras donde no son apropiados
- Ahí es donde entran los formatos XML, JSON, YAML, etc, que son formatos muy útiles para representar objetos

XML: eXtensible Markup Language

Argentina programa 4.0

```xml
<?xml version="1.0" encoding="UTF-8"?>
<carritoCompra>
   <items>
      <item>
         <producto version="1" >
            <nombre>Destornillador</nombre>
            <fechaAlta>03/12/2021</fechaAlta>
            <pesoKg>8</pesoKg>
            <precio vigente="2022">100</precio>
            <stock>10</stock>
         </producto>
         <cantidad>2</cantidad>
      </item>
      <item>
         <producto>
            <nombre>Tuerca</nombre>
            <fechaAlta>20/10/2022</fechaAlta>
            <pesoKg>8</pesoKg>
            <precio>100</precio>
            <stock>9</stock>
         </producto>
         <cantidad>1</cantidad>
      </item>
     </items>
</carritoCompra>
```

- Formato basado en elementos <>
- Los elementos pueden tener texto, otros elementos o menos frecuentemente ambos.
- También pueden tener atributos, que son clave-valor y simples
- Los archivos siempre tienen un único elemento raíz
- Existe más de una forma de representar la misma información. Ej:

```xml
<pesoKg>8</pesoKg>

<peso>
   <magnitud>8</magnitud>
   <unidad>kg</unidad>
</peso>
```

- Forman un ecosistema más complejo, con otras tecnologías como XSD, XSLT, XPATH, XQUERY, que están más allá del alcance del curso.

JSON: Javascript Object Notation

Argentina programa 4.0

```
{
"items":[
        {
                "producto":{
                        "nombre":"Destornillador",
                        "fechaAlta":[2020,20,10],
                        "pesoKg":8,
                        "precio":100,
                        "stock":10
                },
                "cantidad":2
        },
        {
                "producto":{
                        "nombre":"Tuerca",
                        "fechaAlta":[2022,12,03],
                        "pesoKg":8,
                        "precio":100,
                        "stock":9
                },
                "cantidad":1
        }

]
}
```

- Formato clave-valor, con varias similaridades respecto a XML
- Se representan objetos o diccionarios con {} y arrays con []
- Las claves siempre van entre comillas " (notar que NO están en diagonal)
- Formato más reducido, repite menos información que el XML, pero también describe un poco menos

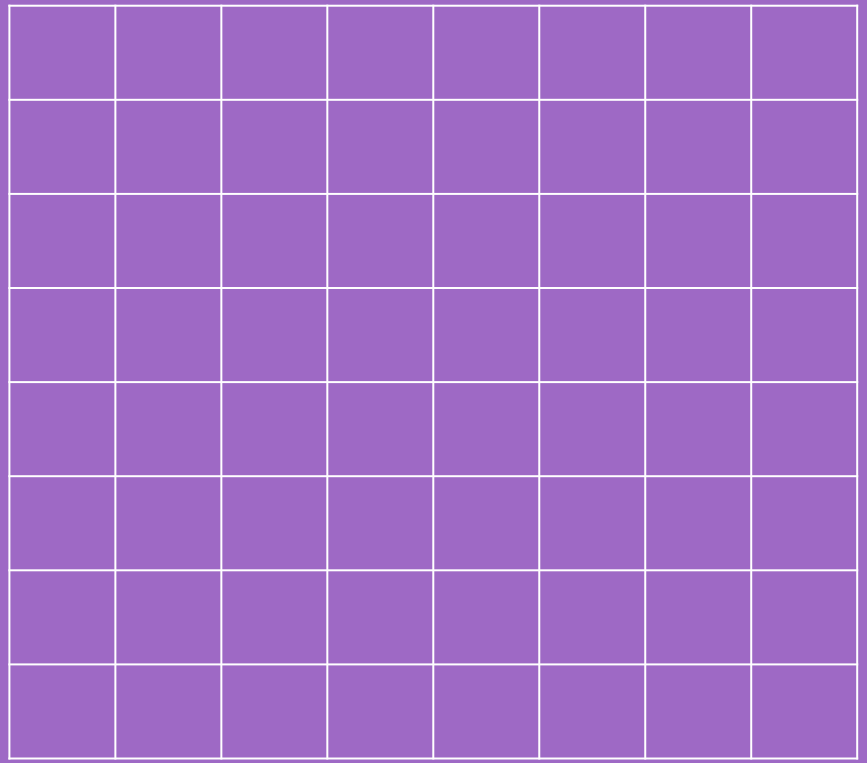
Librerías Serialización / Parseo

Argentina programa 4.0

Estos son 2 términos importantes:

- “Parseo” es un término utilizado para referirse al proceso de transformar un conjunto de bytes o texto a una estructura de datos útil para nosotros. El caso más común de eso es tomar un texto, por ejemplo en formato texto/json y pasarlo a instancias de objetos de nuestro dominio. A veces se usa la palabra Deserializar como sinónimo
- El término serializar proviene de tomar alguna estructura de datos de nuestro programa y transformarla en algún formato útil, ya sea para transmitirlo o para almacenarlo. Por ejemplo pasar un conjunto de objetos a XML
- Java a veces usa serializar/deserializar como una forma particular de hacerlo, que es de forma binaria. Es decir se pasa una serie de objetos a un archivo, pero en un formato que a diferencia de json o xml no puede ser leído

Librerías Serialización / Parseo



Jackson

Argentina programa 4.0

Es un conjunto de herramientas de procesamiento de datos para Java, que incluye la biblioteca de serialización/parseo JSON, la biblioteca de mapeo de objetos a JSON y viceversa y módulos de formato de datos adicionales para procesar datos codificados en XML, YAML, etc (https://github.com/FasterXML/jackson)

```
# En pom.xml
<dependency>
    <groupId>com.fasterxml.jackson.core</groupId>
    <artifactId>jackson-databind</artifactId>
    <version>2.13.3</version>
</dependency>
```

Jackson - Serialización

Argentina programa 4.0

```
class Producto {
      private String nombre;
      private int stock;
      private float precio;


public static void main(String [] args) throws JsonProcessingException {
      ObjectMapper objectMapper = new ObjectMapper();
      Producto prod = new Producto("Destornillador", 10,1.4f);
      String jsonText = objectMapper.writeValueAsString( prod);
      System.out.println(jsonText);
      //{"nombre":"Destornillador","stock":10,"precio":1.4}
}
```

Es importante declarar getters y setters, sino no guarda el atributo

Jackson - Parseo

Argentina programa 4.0

```
class Producto {
	private String nombre;
	private int stock;
	private float precio;
```

Es importante declarar getters y setters, sino no lee el atributo. Tambien, se necesita un constructor vacío

```
String json = "{\"nombre\":\"Destornillador\",\"stock\":10,\"precio\":1.4}";
Producto prod2 = objectMapper.readValue(json,Producto.class);
System.out.println(prod2);
//Producto [nombre=Destornillador, stock=10, precio=1.4]
```

Jackson - Parseo a un Map

Argentina programa 4.0

```
import com.fasterxml.jackson.core.JsonProcessingException;
import com.fasterxml.jackson.core.type.TypeReference;
import com.fasterxml.jackson.databind.ObjectMapper;
//----------------------
String json2 = "{\"nombre\":\"Destornillador\",\"stock\":10,\"precio\":1.4}";
Map<String, Object> map
= objectMapper.readValue(json2, new TypeReference<Map<String,Object>>(){});
System.out.println(map);
//{nombre=Destornillador, stock=10, precio=1.4}
```

Jackson - Relaciones más complejas

Argentina programa 4.0

```
Persona ramon = new Persona("Ramon", "Perea");
CarritoCompra carrito = new CarritoCompra(ramon);
Producto prod = new Producto("destornillador", 11, 10f);
ItemCarrito item = new ItemCarrito();
item.setCantidad(3);
item.setProducto(prod);
carrito.agregarItem(item);
prod = new Producto("tuerca", 1100, 0.01f);
item = new ItemCarrito();
item.setCantidad(3);
item.setProducto(prod);
carrito.agregarItem(item);
ObjectMapper objectMapper = new ObjectMapper();
String jsonText =
objectMapper.writerWithDefaultPrettyPrinter(
     ).writeValueAsString( carrito);
System.out.println(jsonText);
```

```
{
"items" : [ {
     "producto" : {
     "nombre" : "destornillador",
     "stock" : 11,
     "precio" : 10.0
     },
     "cantidad" : 3
     }, {
     "producto" : {
     "nombre" : "tuerca",
     "stock" : 1100,
     "precio" : 0.01
     },
     "cantidad" : 3
     }]
}
```

Jackson - Relaciones más complejas

Argentina programa 4.0

Cuando se quieren ignorar atributos para no serializar se puede usar sobre la declaración del atributo la anotación @JsonIgnore. Cuidado con las relaciones bidireccionales, que pueden romper la serialización.

Por ejemplo si el ítem conociera al carrito, el atributo tendría que ser ignorado:

```java
public class ItemCarrito {

    private Producto producto;

    private int cantidad;

    @JsonIgnore

    private CarritoCompra carrito;
```

```
{
"items" : [ {
      "producto" : {
      "nombre" : "destornillador",
      "stock" : 11,
      "precio" : 10.0
      },
      "cantidad" : 3
      }, {
      "producto" : {
      "nombre" : "tuerca",
      "stock" : 1100,
      "precio" : 0.01
      },
      "cantidad" : 3
      }]
}
```

Referencias

Argentina programa 4.0

- http://www.javalearningacademy.com/streams-in-java-concepts-and-usage/
- https://www.baeldung.com/jackson-object-mapper-tutorial
- https://www.w3schools.com/xml/
- https://www.w3schools.com/js/js_json_intro.asp

Argentina
programa
4.0

Gracias!