TEMA 2: Programación de comunicaciones en red

unir FORMACIÓN PROFESIONAL PROEDUCA

En esta lección perseguimos los siguientes objetivos:

1. Conocer algunos formatos ligeros que serán útiles para la transferencia de información dentro de los sistemas distribuidos.
2. Conocer el formato y sintaxis de los documentos XML.
3. Conocer el formato y sintaxis de los documentos JSON.

Formato de los documentos XML

XML (*Extended Markup Language*) provee un formato ligero para el intercambio de datos, muy interesante para la transmisión de datos en aplicaciones distribuidas.

Tiene su origen en SGML (*Standard Generalized Markup Language*).

Los documentos SGML siguen un esquema jerárquico, compuesto por etiquetas con apertura y cierre que tienen la siguiente estructura:

<etiqueta> contenido </etiqueta>. Cada etiqueta puede portar otras subetiquetas, construyendo así la jerarquía.

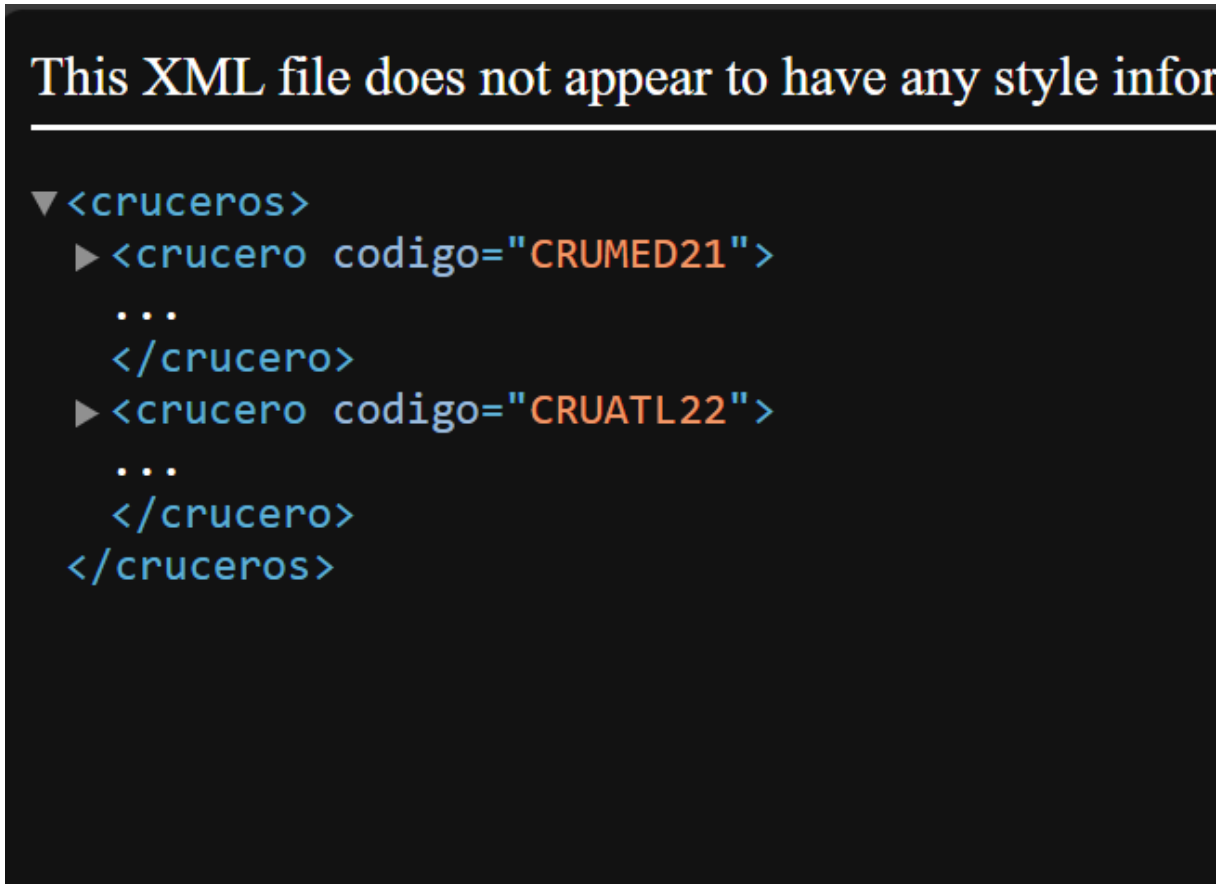
```
<?xml version="1.0" encoding="UTF-8"?>
<cruceros>
<crucero codigo="CRUMED21">
<destino>Mediterraneo (Grecia, Italia)</destino>
<detalles>
<cia>Costa cruceros</cia>
<dias>6 días</dias>
<fechaSalida>2018-12-26</fechaSalida>
</detalles>
<escalas>
<escala dia="1">
<parada>Venecia</parada>
<llegada></llegada>
<salida>18:00</salida>
</escala>
```

La primera línea del documento (**cabecera**), es la que identifica el documento como tipo XML
y
donde se especi ca la versión, tipo de codificación, etc. Luego va la **apertura del nodo raíz,**
que encierra el resto de los nodos.

```
<?xml version="1.0" encoding="UTF-8"?>
<cruceros>
.....
</cruceros>
```

This XML file does not appear to have any style information associated with it. The document tree is shown below.


```
<cruceros>
  <crucero codigo="CRUMED21">
    <destino>Mediterraneo (Grecia, Italia)</destino>
    <detalles>
      <cia>Costa cruceros</cia>
      <dias>6 días</dias>
      <fechaSalida>2018-12-26</fechaSalida>
    </detalles>
    <escalas>
      <escala dia="1">
        <parada>Venecia</parada>
        <llegada/>
        <salida>18:00</salida>
      </escala>
      <escala dia="2">
        <parada>Navegación</parada>
        <llegada/>
        <salida/>
      </escala>
      <escala dia="3">
        <parada>Agostini</parada>
        <llegada>7:00</llegada>
        <salida>14:00</salida>
      </escala>
      <escala dia="4">
        <parada>Santorini</parada>
        <llegada>9:00</llegada>
        <salida>20:00</salida>
      </escala>
      <escala dia="5">
        <parada>Bari</parada>
        <llegada>8:00</llegada>
        <salida>14:00</salida>
      </escala>
      <escala dia="6">
        <parada>Venecia</parada>
        <llegada>8:30</llegada>
        <salida/>
      </escala>
    </escalas>
  </crucero>
```


This XML file does not appear to have any style infor
<cruceros>
<crucero codigo="CRUMED21">
...
</crucero>
<crucero codigo="CRUATL22">
...
</crucero>
</cruceros>

Java dispone de una tecnología muy simple para convertir objetos a formato XML a partir de la librería JAXB, localizada en *javax.xml.bind*.

JAXB son las iniciales de **Java XML API Binding** y nos permite convertir objetos a formato XML. También podemos realizar la operación inversa, es decir, convertir documentos XML en objetos Java.

Necesitamos trabajar con JAXB (Java Architecture for XML Binding) ya no está incluido en el JDK desde Java 9. Para solucionarlo en IntelliJ, debes agregar la dependencia de JAXB manualmente

Agregar JAXB como dependencia en Maven

Si usas **Maven**, agrega esto en tu pom.xml dentro de <dependencies>:

```
<dependency> <groupId>jakarta.xml.bind</groupId>
<artifactId>jakarta.xml.bind-api</artifactId>
<version>3.0.1</version> </dependency> <dependency>
<groupId>org.glassfish.jaxb</groupId> <artifactId>jaxb-
runtime</artifactId> <version>3.0.1</version>
</dependency>
```

Agregar manualmente el JAR de JAXB

Si no usas Maven, descarga el JAR desde:

- https://mvnrepository.com/artifact/org.glassfish.jaxb/jaxb-runtime

Luego:

1. En IntelliJ, ve a **File > Project Structure > Libraries**.
2. Haz clic en **+ (Agregar nueva librería) > From JARs**.
3. Selecciona el archivo descargado.

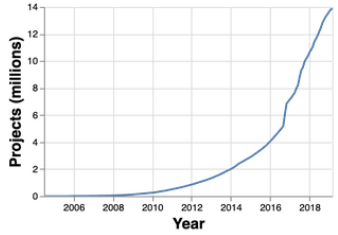
https://mvnrepository.com/artifact/org.glassfish.jaxb/jaxb-runtime

MVN REPOSITORY

Search for groups, artifacts, categories

Search

Indexed Artifacts (50.3M)



Popular Categories

- Testing Frameworks & Tools
- Android Packages
- Logging Frameworks
- JVM Languages
- Java Specifications
- JSON Libraries
- Core Utilities
- Mocking
- Annotation Libraries
- Web Assets
- Language Runtime
- HTTP Clients
- Logging Bridges
- Dependency Injection

Home » org.glassfish.jaxb » jaxb-runtime

JAXB Runtime

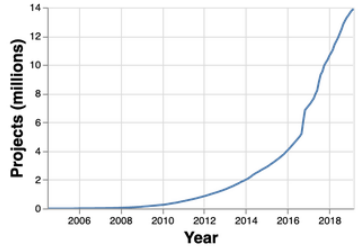
JAXB (JSR 222) Reference Implementation

License	EDL 1.0
Tags	binding glassfish jaxb xml runtime
HomePage	https://eclipse-ee4j.github.io/jaxb-ri/
Ranking	**#196 in MvnRepository (See Top Artifacts)**
Used By	**2,770 artifacts**

Central (41) | Redhat GA (11) | Redhat EA (11) | JBoss 3rd-party (1) | SciJava Public (2) | ECSEC OpenEcard (1) | Payara (1)

	Version	Vulnerabilities	Repository	Usages	Date
4.0.x	4.0.5		Central	360	*Mar 07, 2024*
	4.0.4		Central	237	*Oct 20, 2023*
	4.0.3		Central	137	*Jun 09, 2023*
	4.0.2		Central	174	*Feb 06, 2023*
	4.0.1		Central	98	*Sep 20, 2022*
	4.0.0		Central	54	*Jun 06, 2022*
	4.0.0-M4		Central	12	*Apr 11, 2022*
	4.0.0-M3		Central	12	*Apr 11, 2022*

Indexed Artifacts (50.3M)



Popular Categories

- Testing Frameworks & Tools
- Android Packages
- Logging Frameworks
- JVM Languages
- Java Specifications
- JSON Libraries
- Core Utilities
- Mocking
- Annotation Libraries
- Web Assets
- Language Runtime
- HTTP Clients
- Logging Bridges

Home » org.glassfish.jaxb » jaxb-runtime » 2.3.1

JAXB Runtime » 2.3.1

JAXB (JSR 222) Reference Implementation

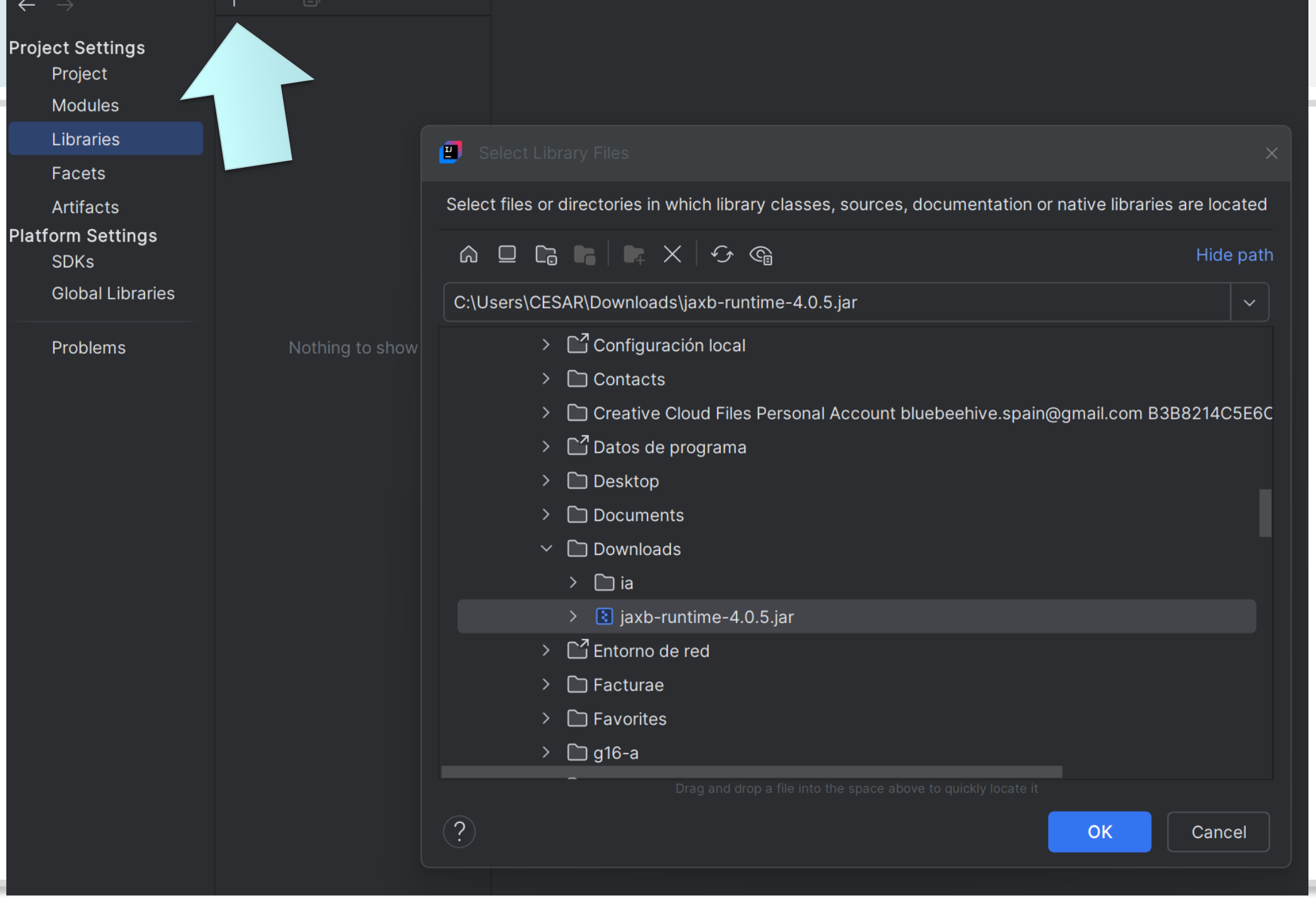
License	CDDL GPL 1.1
Tags	binding glassfish jaxb xml runtime
Date	Oct 02, 2018
Files	pom (7 KB) jar (1.0 MB) View All
Repositories	Central Archive Gael Java.net JCenter OneBusAway Pub Redhat EA SciJava Public SpaceIO WSO2 Public
Ranking	**#196 in MvnRepository (See Top Artifacts)**
Used By	**2,770 artifacts**
Vulnerabilities	**Vulnerabilities from dependencies:** **CVE-2020-15250**

Note: There is a new version for this artifact

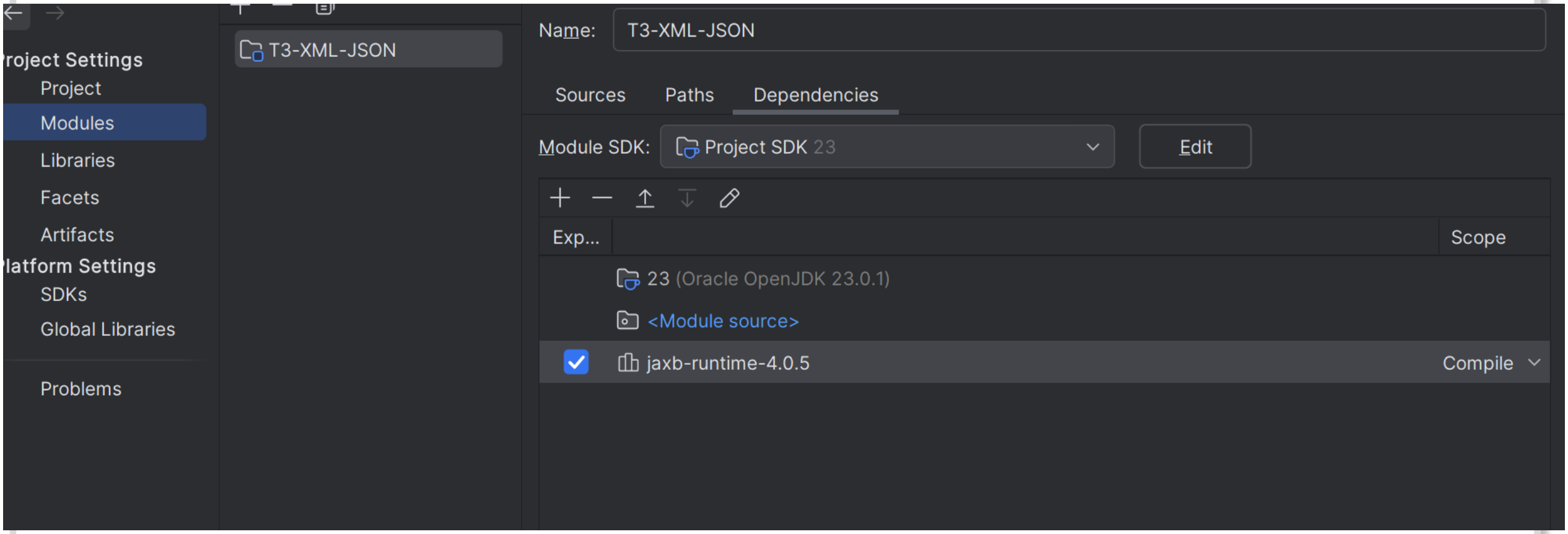
New Version	4.0.5

Maven | Gradle | Gradle (Short) | Gradle (Kotlin) | SBT | Ivy | Grape | Leiningen | Buildr

```
<!-- https://mvnrepository.com/artifact/org.glassfish.jaxb/jaxb-runtime -->
<dependency>
```


Project Settings
Project
Modules
Libraries
Facets
Artifacts
Platform Settings
SDKs
Global Libraries
Problems
Nothing to show
Select Library Files
Select files or directories in which library classes, sources, documentation or native libraries are located
Hide path
C:\Users\CESAR\Downloads\jaxb-runtime-4.0.5.jar
Configuración local
Contacts
Creative Cloud Files Personal Account bluebeehive.spain@gmail.com B3B8214C5E6C
Datos de programa
Desktop
Documents
Downloads
ia
jaxb-runtime-4.0.5.jar
Entorno de red
Facturae
Favorites
g16-a
Drag and drop a file into the space above to quickly locate it
OK
Cancel

Después de cualquiera de estas opciones,
reinicia IntelliJ y prueba de nuevo.

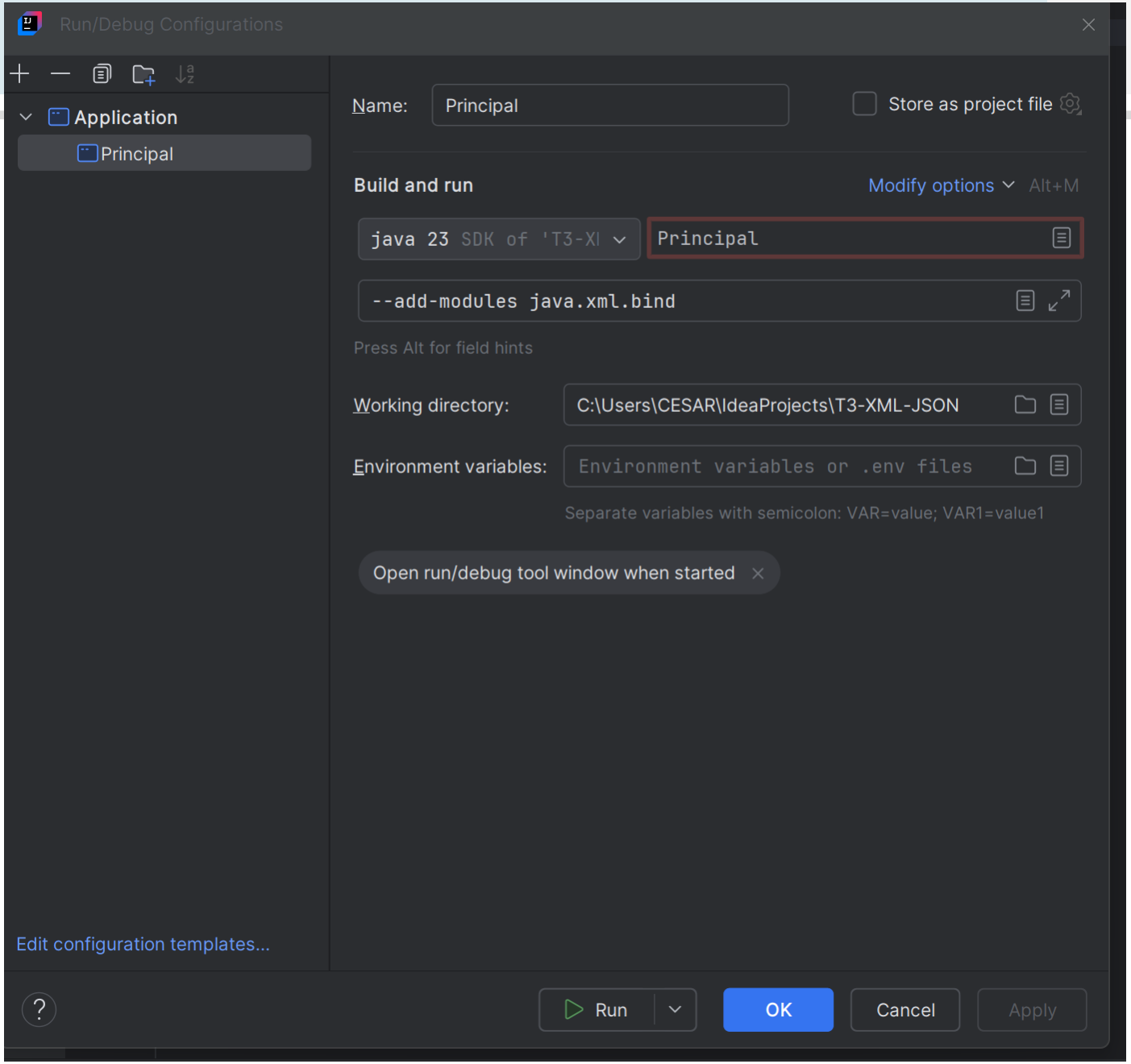

Project Settings
Project
Modules
Libraries
Facets
Artifacts
Platform Settings
SDKs
Global Libraries
Problems
T3-XML-JSON
Name: T3-XML-JSON
Sources
Paths
Dependencies
Module SDK: Project SDK 23
Edit
Exp...
Scope
23 (Oracle OpenJDK 23.0.1)
<Module source>
jaxb-runtime-4.0.5
Compile

Si sigues teniendo problemas con JAXB, puedes probar lo siguiente:

Si usas Java 9 o superior, agrega la JVM opción:

--add-modules java.xml.bind

Para hacerlo en IntelliJ:
1.Ve a Run > Edit Configurations.
2.En VM Options, agrega --add-modules java.xml.bind.


Run/Debug Configurations
Application
Principal
Name:
Principal
Store as project file
Build and run
Modify options
Alt+M
java 23 SDK of 'T3-XI
Principal
--add-modules java.xml.bind
Press Alt for field hints
Working directory:
C:\Users\CESAR\IdeaProjects\T3-XML-JSON
Environment variables:
Environment variables or .env files
Separate variables with semicolon: VAR=value; VAR1=value1
Open run/debug tool window when started
Edit configuration templates...
Run
OK
Cancel
Apply

Tambien puede faltar la API de JAXB (jakarta.xml.bind-api). jaxb-runtime solo proporciona la implementación, pero también necesitas la API.

Agregar jaxb-api-2.3.1

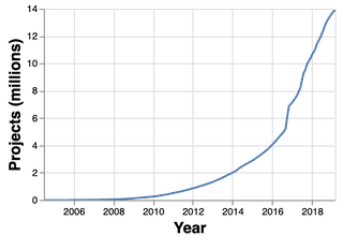
Descarga la API de JAXB desde aquí.

2.Agrega el JAR en IntelliJ:

- Ve a File > Project Structure > Modules > Dependencies.
- Pulsa + > JARs or directories.
- Selecciona el JAR de jaxb-api-2.3.1.jar.
- Asegúrate de que el **Scope** esté en Compile.

https://mvnrepository.com/artifact/javax.xml.bind/jaxb-api/2.3.1

Indexed Artifacts (50.3M)



Popular Categories

Testing Frameworks & Tools
Android Packages
Logging Frameworks
JVM Languages
Java Specifications
JSON Libraries
Core Utilities
Mocking
Annotation Libraries
Web Assets
Language Runtime
HTTP Clients
Logging Bridges
Dependency Injection
XML Processing

Home » com.sun.istack » istack-commons-runtime » 3.0.12

Istack Common Utility Code Runtime » 3.0.12

Istack Common Utility Code Runtime

License	EDL 1.0
Tags	commons runtime
Date	Apr 08, 2021
Files	pom (4 KB) jar (29 KB) View All
Repositories	Central GroovyPlugins Mulesoft Orekit Talend Public USIT
Ranking	**#636 in MvnRepository (See Top Artifacts)**
Used By	**865 artifacts**

Note: There is a new version for this artifact

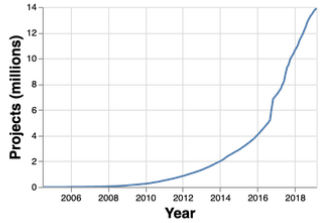
New Version	4.2.0

Maven | Gradle | Gradle (Short) | Gradle (Kotlin) | SBT | Ivy | Grape | Leiningen | Buildr

```
<!-- https://mvnrepository.com/artifact/com.sun.istack/istack-commons-runtime -->
<dependency>
    <groupId>com.sun.istack</groupId>
    <artifactId>istack-commons-runtime</artifactId>
    <version>3.0.12</version>
</dependency>
```

Include comment with link to declaration

Indexed Artifacts (50.3M)



Popular Categories

Testing Frameworks & Tools
Android Packages
Logging Frameworks
JVM Languages
Java Specifications
JSON Libraries
Core Utilities
Mocking
Annotation Libraries
Web Assets
Language Runtime
HTTP Clients
Logging Bridges
Dependency Injection
XML Processing

Home » javax.activation » activation » 1.1.1

JavaBeans(TM) Activation Framework » 1.1.1

The JavaBeans(TM) Activation Framework is used by the JavaMail(TM) API to manage MIME data

License	CDDL 1.0
Categories	Java Specifications
Tags	javax specs standard
HomePage	http://java.sun.com/javase/technologies/desktop/javabeans/ja ...
Date	Oct 23, 2009
Files	pom (644 bytes) jar (67 KB) View All
Repositories	Central AKSW Alfresco Atricore JCenter Sakai SciJava Public Xceptance
Ranking	**#249 in MvnRepository (See Top Artifacts)** **#16 in Java Specifications**
Used By	**2,209 artifacts**

Maven | Gradle | Gradle (Short) | Gradle (Kotlin) | SBT | Ivy | Grape | Leiningen | Buildr

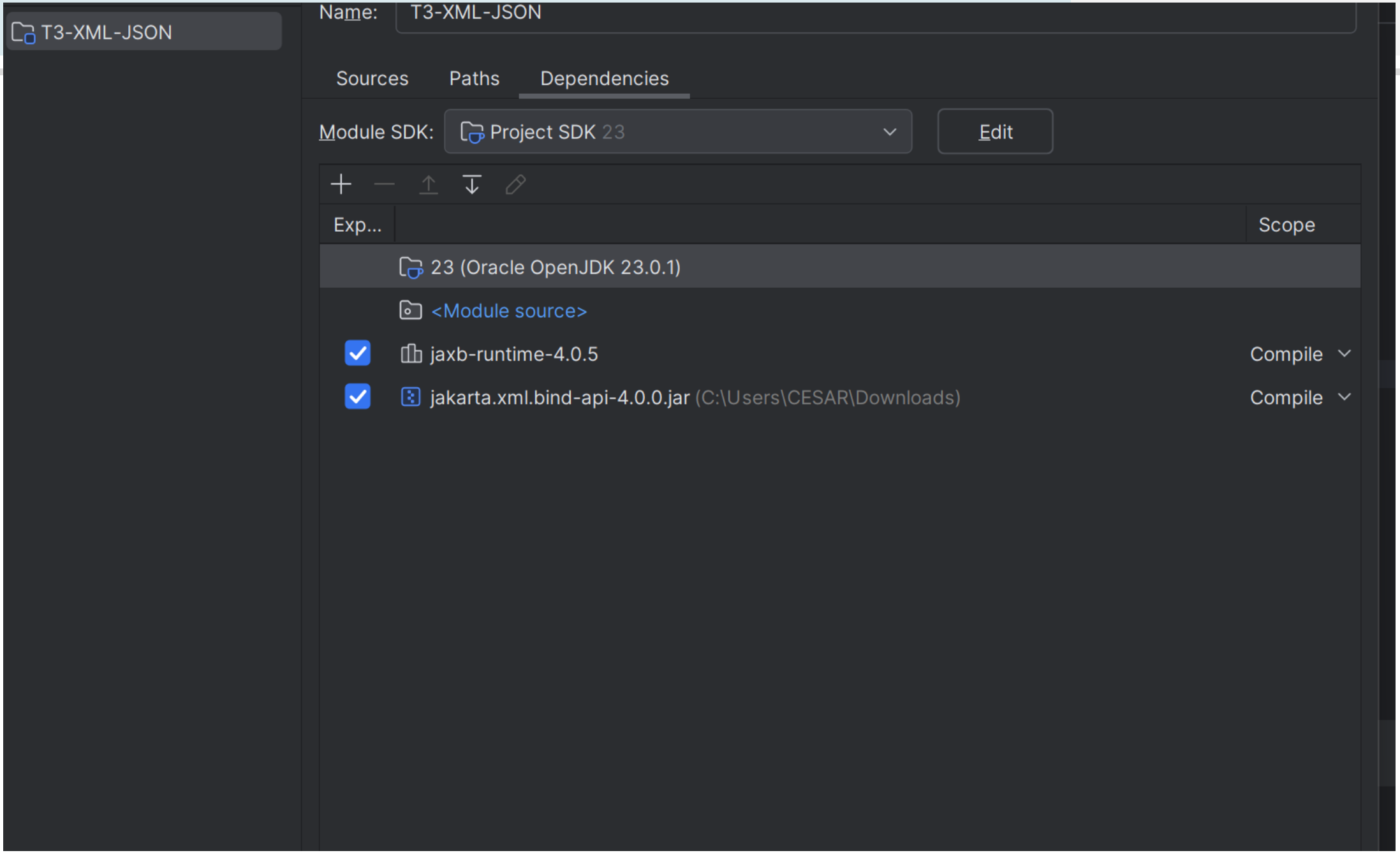
```
<!-- https://mvnrepository.com/artifact/javax.activation/activation -->
<dependency>
    <groupId>javax.activation</groupId>
    <artifactId>activation</artifactId>
    <version>1.1.1</version>
</dependency>
```

☑ Include comment with link to declaration

Tambien puede faltar la API de JAXB (jakarta.xml.bind-api). jaxb-runtime solo proporciona la implementación, pero también necesitas la API.

Puedes descargar la API de JAXB desde este enlace oficial en **Maven Repository**:
Descargar jakarta.xml.bind-api-4.0.0.jar
1.Entra en la página.
2.Haz clic en el botón **Download (jar)**.
3.Luego, agrégalo a IntelliJ en File > Project Structure > Modules > Dependencies. (Jars or Directories)

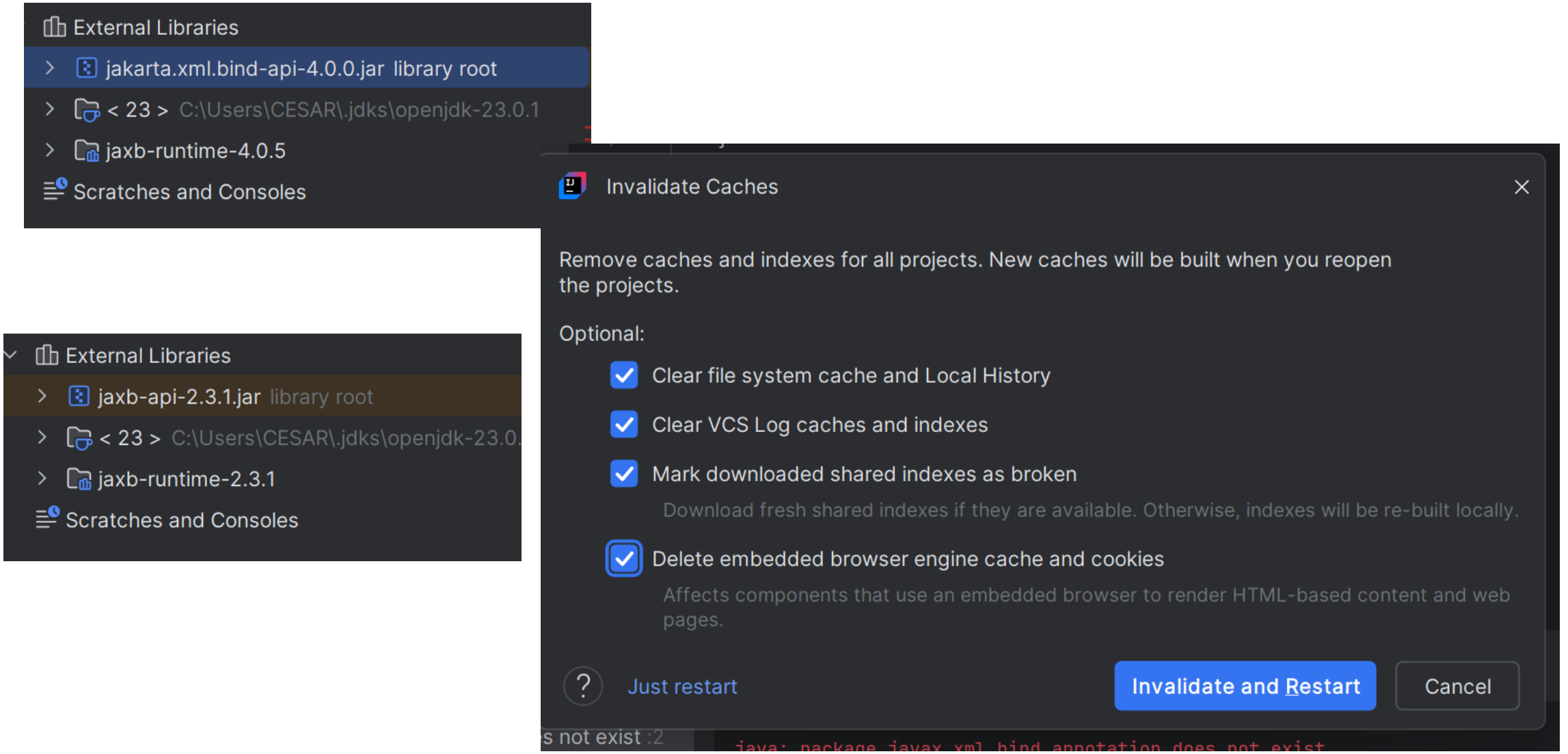
https://mvnrepository.com/artifact/jakarta.xml.bind/jakarta.xml.bind-api/4.0.0


T3-XML-JSON
Name: T3-XML-JSON
Sources
Paths
Dependencies
Module SDK: Project SDK 23
Edit
Exp...
Scope
23 (Oracle OpenJDK 23.0.1)
<Module source>
jaxb-runtime-4.0.5
Compile
jakarta.xml.bind-api-4.0.0.jar (C:\Users\CESAR\Downloads)
Compile

Si usas Maven, agrega estas dependencias en pom.xml:

```
<dependencies>    <dependency>
<groupId>jakarta.xml.bind</groupId>
<artifactId>jakarta.xml.bind-api</artifactId>
<version>4.0.0</version>    </dependency>
<dependency>
<groupId>org.glassfish.jaxb</groupId>
<artifactId>jaxb-runtime</artifactId>
<version>4.0.5</version>
</dependency></dependencies>
```

Luego, **refresca el proyecto** (Maven > Reload Project).


External Libraries
jakarta.xml.bind-api-4.0.0.jar library root
< 23 > C:\Users\CESAR\.jdks\openjdk-23.0.1
jaxb-runtime-4.0.5
Scratches and Consoles
External Libraries
jaxb-api-2.3.1.jar library root
< 23 > C:\Users\CESAR\.jdks\openjdk-23.0.
jaxb-runtime-2.3.1
Scratches and Consoles
Invalidate Caches
Remove caches and indexes for all projects. New caches will be built when you reopen the projects.
Optional:
Clear file system cache and Local History
Clear VCS Log caches and indexes
Mark downloaded shared indexes as broken
Download fresh shared indexes if they are available. Otherwise, indexes will be re-built locally.
Delete embedded browser engine cache and cookies
Affects components that use an embedded browser to render HTML-based content and web pages.
Just restart
Invalidate and Restart
Cancel

Se trata de una clase normal y corriente, pero debe utilizar las anotaciones necesarias para que el sistema pueda interpretar la forma en la que deberá construir el documento XML.

Además, **será imprescindible que la clase cuente con un constructor sin argumentos.** Con la anotación ***@XmlRootElement(name = "persona")***, estamos especi cando el nombre del elementos raíz.

Anotamos con ***@XmlElement*** los métodos *get* correspondientes a las propiedades del objeto que deseamos importar.

Anotamos con ***@XmlAttribute*** el método *getIdPersona()* porque deseamos utilizar la propiedad *idPersona* como atributo de la etiqueta raíz (elemento persona).

la clase ***Principal*** que construye un objeto *persona* y lo convierte a formato XML.

Nuestra clase principal necesitará contar con dos importantes objetos, cuyas clases pertenecen a la librería JAXB.

1. Un objeto de la clase ***JAXBContext*** al que llamaremos *contexto* y que estará asociado a la clase *Persona.* Será utilizado por el objeto *Marshaller* para localizar información respecto a la estructura del objeto a convertir.

2. Un objeto de la clase ***Marshaller*** al que llamaremos *m* que servirá para realizar la conversión.

```
contexto = JAXBContext.newInstance(Persona.class);
```

Obtiene el contexto asociado a la clase *Persona*. Provoca una excepción de tipo *JAXBException* si la clase *Persona* no cumple los requisitos para la conversión a XML, es decir, contener las anotaciones necesarias y contar con un constructor sin argumentos.

```
m = contexto.createMarshaller();
```

Obtiene el objeto *Marshaller* asociado al contexto.

```
m.setProperty(Marshaller.JAXB_FORMATTED_OUTPUT, true);
```

Establecer la propiedad *JAXB_FORMATTED_OUTPUT* con el valor *true* permite que en la conversión a formato XML se incluyan retornos de carro e indentación (sangrado del texto). Prueba a ejecutar el programa con los valores *true* y *false* para ver la diferencia.

```
Persona p = new Persona(1, "Homer", "Simpson", 48);
m.marshal(p, System.out);
```

Crea una objeto *persona*, lo convierte a XML y devuelve el resultado a la consola (*System.out*).

Si lo que deseamos es convertir un documento XML en un objeto java de la clase *Persona*, necesitaremos **dos objetos principales**:

- Un nuevo objeto ***JAXBContext*** que conocerá la estructura de la clase *Persona*.
- Un objeto de la clase ***Unmarshaller*** que se encargará de exportar el archivo XML *Homer.xml* en un nuevo objeto *Persona*.

A continuación complicaremos un poco el formato del futuro documento XML.

Y lo haremos exportando, no un único objeto *Persona*, sino una colección de tipo *ArrayList* de objetos *Persona*.

¿Qué es JSON?

JSON viene de las iniciales *JavaScript Object Notation*, ya que se trata del formato que utiliza JavaScript para la construcción de objetos.

Sintaxis

Para crear un documento JSON debemos seguir las siguientes **reglas**:

- Los datos deben estar separados por comas.
- Los datos se pondrán con la estructura *"clave:valor"*.
- Tanto las *claves* como los *valores* irán entre comillas.
- Los objetos JSON se definen por llaves “{}”.
- Los *arrays* van con corchetes [].
- Tanto los objetos como los *arrays* pueden contener otros objetos o *arrays*.

```
{
        "nombre":"Homer",
        "edad":"45",
        "genero":"masculino",
        "email":"homer@dominio.com",
        "localidad":"Springfield",
        "telefono":"91000000"
}
```

Objeto JSON simple. Cada objeto está delimitados por los carácteres { y }.

```
{
        "nombre": "Homer",
        "edad": "45",
        "genero": "masculino",
        "email": "homer@dominio.com",
        "localidad": "Springfield",
        "telefono": "91000000",
        "aficiones": ["beber cerveza", "comer", "divertirse"]
}
```

Hemos incluido el campo aficiones que es un *array*.

```
{
        "nombre": "Homer",
        "edad": "45",
        "genero": "masculino",
        "email": "homer@dominio.com",
        "domicilio": {
                "calle": "Percebe",
                "numero": 7,
                "piso": 2,
                "puerta": "A"
        },
        "localidad": "Springfield",
        "telefono": "91000000",
        "aficiones": ["beber cerveza", "comer", "divertirse"]
}
```

Campo *domicilio* formado por un objeto compuesto por: *calle*, *numero*, *piso* y *puerta*.

Cada campo en un objeto JSON está compuesto por un par *clave: valor*.

```
"nombre":"valor"
```

Para definir un objeto utilizamos las llaves {}.

```
{
        "nombre": "valor"
}
```

Para definir varios elementos los separaremos por comas ",".

```
{
        "nombre": "Juan",
        "apellidos": "Gonzalez"
}
```

JSON permite la definición de *arrays* con el uso de los corchetes [ ].

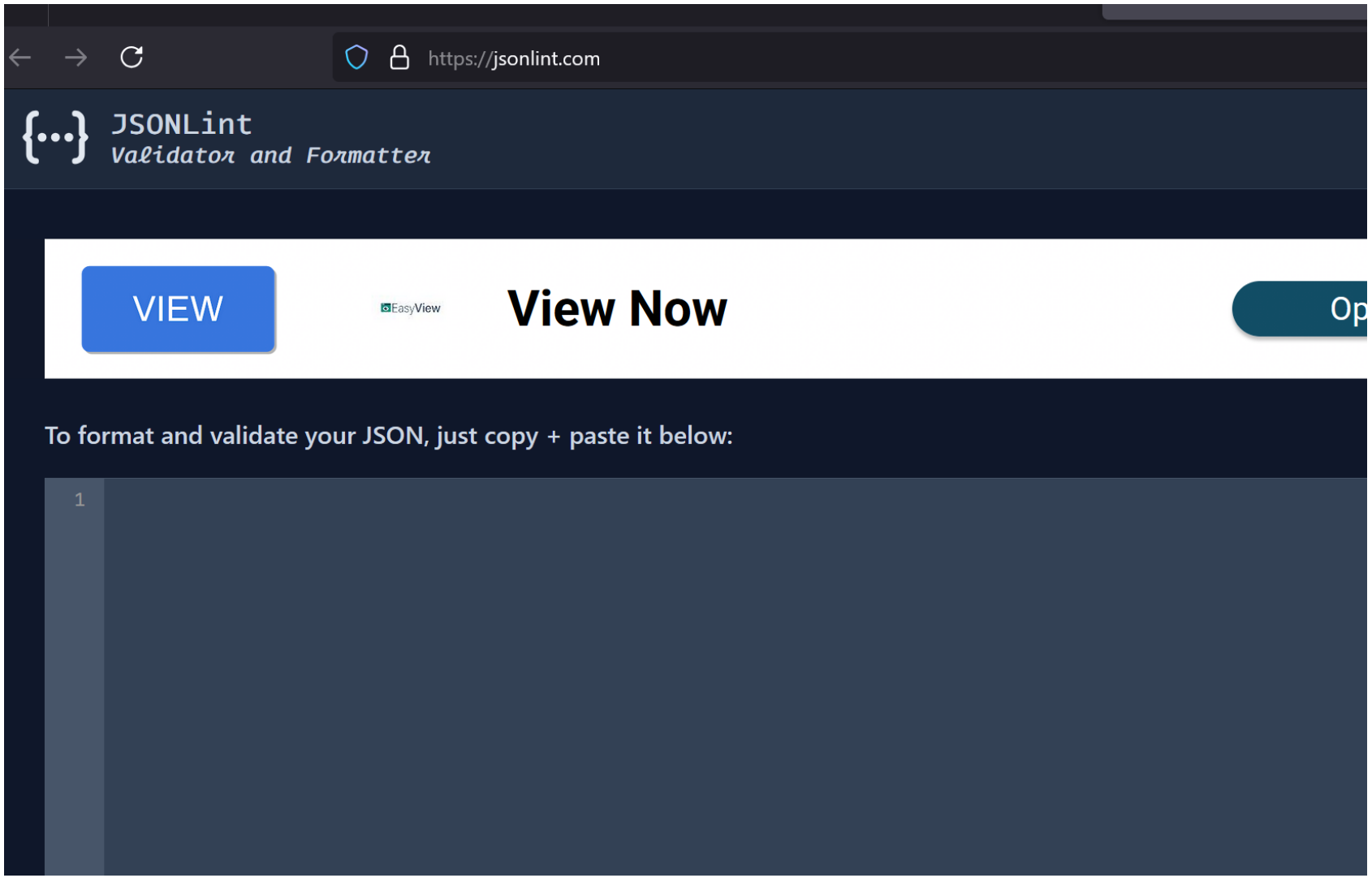
```
{
        "nombre": "juan",
        "edad": 30,
        "coches": ["Ford", "BMW", "Fiat"]
}
```

```
{
     "empresa": "Ferreteria Carmela",
     "domicilio": "C/ Luna, 45",
     "empleados": [{
          "nombre": "Juan",
          "apellidos": "Gonzalez"
     }, {
          "nombre": "María",
          "apellidos": "Alonso"
     }, {
          "nombre": "José",
          "apellidos": "amohedo"
     }]
}
```

Este objeto JSON representa una empresa con los campos empresa, domicilio y empleados, compuesto por un *array* de objetos cada uno de los cuales contiene los datos de un trabajador.

En este apartado te mostraremos los tipos de datos que son soportados por JSON:

- ***string:*** cadenas de texto.
- ***number:*** datos numéricos.
- ***object:*** objetos JSON.
- ***arrays:*** *arrays*, vectores, etc.
- ***boolean:*** datos booleanos (*true*, *false*).
- ***null:*** valores nulos.



https://jsonlint.com/

Crea un fichero JSON con estos datos:

Clave	Tipo de dato	Valor de ejemplo
`"nombre"`	String	`"Juan Pérez"`
`"edad"`	Number	`30`
`"direccion"`	Object (JSON anidado)	`{"calle": "Av. Siempre Viva", "numero": 742}`
`"telefonos"`	Array	`["123-456-789", "987-654-321"]`
`"casado"`	Boolean	`false`
`"hijos"`	Null	`null`

```
{
  "nombre": "Juan Pérez",      // String
  "edad": 30,                  // Number
  "direccion": {               // Object
    "calle": "Av. Siempre Viva",
    "numero": 742
  },
  "telefonos": [               // Array
    "123-456-789",
    "987-654-321"
  ],
  "casado": false,             // Boolean
  "hijos": null                // Null
}
```

Exportar objetos java a formato JSON no puede ser más fácil con la librería *GSON* de Google.

Volveremos a utilizar las clases *Persona* y *GrupoPersonas* para exportar los datos de la familia Simpson a formato JSON.

https://repo1.maven.org/maven2/com/google/code/gson/gson/2.8.1/

Agregar el JAR en IntelliJ

1.Ve a File > Project Structure > Modules > Dependencies.
2.Haz clic en **+** y selecciona **"JARs or directories"**.
3.Busca el archivo **gson-2.8.1.jar** que descargaste y selecciónalo.
4.Asegúrate de que el **Scope** esté en **Compile**.
5.Pulsa **Apply** y luego **OK**.

MAVEN
Abre el archivo pom.xml.
Agrega esta dependencia dentro de <dependencies>:

<dependency>
<groupId>com.google.code.gson</groupId>
<artifactId>gson</artifactId>
<version>2.8.1</version>
</dependency>

com/google/code/gson/gson/2.8.1

../		
gson-2.8.1-javadoc.jar	2017-05-31 01:39	262469
gson-2.8.1-javadoc.jar.asc	2017-05-31 01:39	473
gson-2.8.1-javadoc.jar.asc.md5	2017-05-31 01:39	32
gson-2.8.1-javadoc.jar.asc.sha1	2017-05-31 01:39	40
gson-2.8.1-javadoc.jar.md5	2017-05-31 01:39	32
gson-2.8.1-javadoc.jar.sha1	2017-05-31 01:39	40
gson-2.8.1-sources.jar	2017-05-31 01:39	147468
gson-2.8.1-sources.jar.asc	2017-05-31 01:39	473
gson-2.8.1-sources.jar.asc.md5	2017-05-31 01:39	32
gson-2.8.1-sources.jar.asc.sha1	2017-05-31 01:39	40
gson-2.8.1-sources.jar.md5	2017-05-31 01:39	32
gson-2.8.1-sources.jar.sha1	2017-05-31 01:39	40
gson-2.8.1.jar	2017-05-31 01:39	232620
gson-2.8.1.jar.asc	2017-05-31 01:39	473
gson-2.8.1.jar.asc.md5	2017-05-31 01:39	32
gson-2.8.1.jar.asc.sha1	2017-05-31 01:39	40
gson-2.8.1.jar.md5	2017-05-31 01:39	32
gson-2.8.1.jar.sha1	2017-05-31 01:39	40
gson-2.8.1.pom	2017-05-31 01:39	1893
gson-2.8.1.pom.asc	2017-05-31 01:39	473
gson-2.8.1.pom.asc.md5	2017-05-31 01:39	32
gson-2.8.1.pom.asc.sha1	2017-05-31 01:39	40
gson-2.8.1.pom.md5	2017-05-31 01:39	32
gson-2.8.1.pom.sha1	2017-05-31 01:39	40

MUCHAS GRACIAS POR VUESTRA ATENCIÓN!