El uso de Servlet JSON es mucho más habitual de lo que parece ya que muchas personas siguen teniendo que desplegar aplicaciones muy muy clásicas que tienen más de una década de ciclo de vida . En ese caso nos encontramos que necesitan para gestionar temas puntuales de Ajax , enviar datos en JSON desde un Servlet a una página HTML .  Lo más sencillo en este caso es partir de un proyecto de Maven que sea de tipo WAR y añadirnos las dependencias de JackSON.

```
<dependency>
    <groupId>com.fasterxml.jackson.core</groupId>
    <artifactId>jackson-databind</artifactId>
    <version>2.12.4</version>

  </dependency>
```

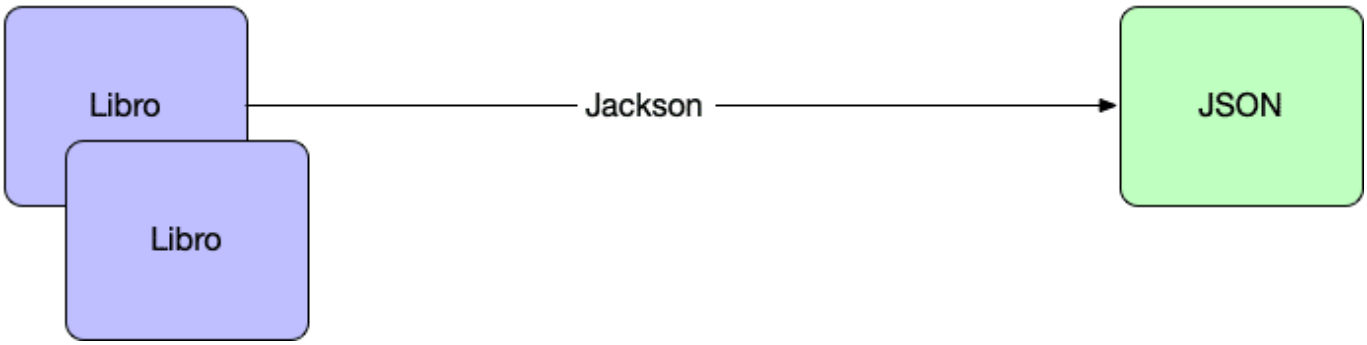
Al añadir esta dependencia Maven nos añadirá las otras librerías requeridas que no son ni más ni menos que  Jackson core y jackson-annotation por si hay alguna persona que no pueda usar Maven. Una vez que tenemos las librerías a nuestra disposición debemos crear un Servlet Sencillo que nos genere una lista de Objetos en formato JSON para ello usaremos la clase Libro.

```
package com.arquitecturajava.negocio;


public class Libro {

  private String isbn;
  private String titulo;
  private String autor;
  public String getIsbn() {
    return isbn;
  }
  public void setIsbn(String isbn) {
```

```
  this.isbn = isbn;
}
public String getTitulo() {
  return titulo;
}
public void setTitulo(String titulo) {
  this.titulo = titulo;
}
public String getAutor() {
  return autor;
}
public void setAutor(String autor) {
  this.autor = autor;
}
public Libro(String isbn, String titulo, String autor) {
  super();
  this.isbn = isbn;
  this.titulo = titulo;
  this.autor = autor;
}
public Libro(String isbn) {
  super();
  this.isbn = isbn;
}
@Override
public int hashCode() {
  final int prime = 31;
  int result = 1;
  result = prime * result + ((isbn == null) ? 0 : isbn.hashCode());
  return result;
}
```

```
  @Override
  public boolean equals(Object obj) {
    if (this == obj)
      return true;
    if (obj == null)
      return false;
    if (getClass() != obj.getClass())
      return false;
    Libro other = (Libro) obj;
    if (isbn == null) {
      if (other.isbn != null)
        return false;
    } else if (!isbn.equals(other.isbn))
      return false;
    return true;
  }
}
```

Una vez tenemos la clase de Negocio es cuestión de crear un Servlet y apoyarnos en la librería de Jackson para convertir  una lista de Objetos en JSON:

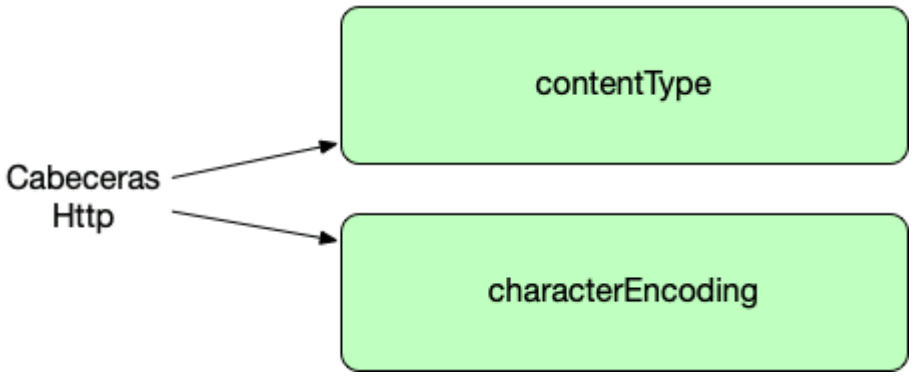


# Servlet JSON y cabeceras

```
package com.arquitecturajava.viejo;

import java.io.IOException;
import java.io.PrintWriter;
import java.util.Arrays;

import javax.servlet.ServletException;
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;

import com.arquitecturajava.negocio.Libro;
import com.fasterxml.jackson.databind.ObjectMapper;

/**
 * Servlet implementation class ServletAjax2
 */
@WebServlet("/ServletAjax2")
public class ServletAjax2 extends HttpServlet {
  private static final long serialVersionUID = 1L;
  protected void doGet(HttpServletRequest request, HttpServletResponse
response) throws ServletException, IOException {
    Libro l= new Libro("1a","java","pedro");
    Libro l2= new Libro("2a","java","pedro");
    ObjectMapper mapeador= new ObjectMapper();
    PrintWriter pw=response.getWriter();
    response.setContentType("application/json");
    response.setCharacterEncoding("UTF-8");
```

```
    pw.print(mapeador.writeValueAsString(Arrays.asList(l,l2)));
  }


}
```

Como podemos ver no solo se trata de usar Jackson a la hora de convertir en este caso una lista de objetos a JSON sino que también implica  asignar las cabeceras HTTP de contentType y characterEncoding que facilite a los clientes leer de forma directa esta información:

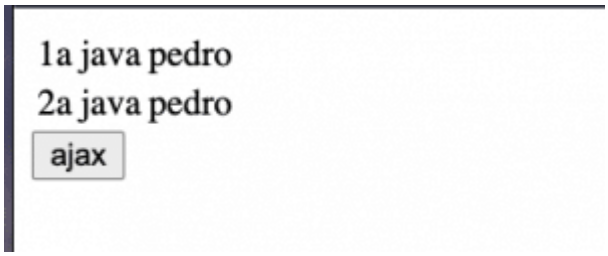



Para acceder la lista de datos nos valdría con hacer algo sencillo a nivel de jQuery:

```
$(document).ready(function() {

    $("#boton").click(function() {

      $.get("../ServletAjax", function(datos) {
        let lista=datos.split(",");
        for (let i=0;i<lista.length;i++) {
          $("#mitabla").append(`<tr><td>${lista[i]}</td></tr>`);
        }
      })
```

```
    })

});
```

Esto nos lo imprimirá en la consola:




Ya disponemos de un Servlet que nos generar JSON y nos permite integrar de forma rápida peticiones Ajax y JSON en aplicaciones clásicas:

## Otros artículos relacionados

1. POJO to JSON y JSON to POJO
2. REST DTO y JSON Arquitecturas Web y objetos
3. JSON API y Arquitecturas REST
4. JAX-RS Client y JSON
5. REST API Design y simplicidad