New Maven proyect

Next

Artifactm id: maven- archetype -webapp || versión: 1.4

Ir a <https://mvnrepository.com/artifact/jakarta.servlet/jakarta.servlet-api/6.0.0>

Rellenar con los datos

Server elegimos tomcat 10.1

Inicamos tomcat

Le damos a start

Run in server

Elegimos tomcat

Ok

Se abrira en el buscador

Empaquetar, es necesario para gestionar bien la aplicaion/proyecto

Nos creamos un servlet

Creamos dentro del método service que traga todo

Dentro del paquete de ejemplos

Para enviar una respuesta del servidor al cliente es response

response .setcontentType(“text/plain”); -> para que devuelva texto plano y no un archivo

response.getWriter().append(“HOLA MUNDO”); -> responde HOLA MUNDO

creamos otro para que devuelva un archivo html

response .setcontentType(“text/html”);

response.getWriter().append(“<HTML>”);

response.getWriter().append(“<BODY>”);

response.getWriter().append(“<H1>”);

response.getWriter().append(“HOLA MUNDO”);

response.getWriter().append(“</H1>”);

response.getWriter().append(“</BODY>”);

response.getWriter().append(“</HTML>”);

response.getWriter().close ();

Otro servlet con init , service y Destroyer

El server va a extender de http server, que este extiende de genery server, que de aquí tiramos init, service, Destroyer, y del http extendemos las peticiones

En el init temos el super donde llamamos al init(config) del genery server

El método service ba tener las diferentes peticiones del servidor cliente, recoge cualquier tio de posición independientemente si le viene por donde sea(do put, do get, etc), entonces no es buena practica

Se le pasa un serverrequest y sereverreponse, para que pueda recibir y responder , y siempre es asi

Para pillar la respues del cliente al servidor se