**1. Metodi HTTP e Java**

In un’applicazione Java che utilizza il protocollo HTTP, ogni verbo (GET, POST, PUT, DELETE, ecc.) corrisponde a un metodo. Questo permette di separare il modo in cui si gestisce ogni tipo di richiesta:

* GET → Metodo doGet della Servlet
* POST → Metodo doPost della Servlet

In pratica, quando arriva una richiesta HTTP, la Servlet invoca il metodo appropriato (es. doGet) per gestire la richiesta specifica.

**Domande:**

* Quali sono i metodi principali del protocollo HTTP?
* A cosa serve il metodo doGet e quando viene utilizzato?

**2. Configurazione di un Application Server**

Per creare una web application in Java, è necessario un **Application Server (AS)**, configurato per gestire le richieste delle applicazioni. Un AS, come **WildFly** o **Tomcat**, accetta le richieste sulla "porta applicativa" (es. 8080) e le instrada alle risorse richieste (Servlet, JSP). La porta applicativa funge da punto d’accesso e permette al server di ascoltare e rispondere alle richieste HTTP dei client.

**Domande:**

* Cos’è la porta applicativa e a cosa serve?
* Quali sono alcuni esempi di application server compatibili con Java?

**3. Differenze tra WildFly e Tomcat**

* **WildFly** richiede che sia configurato il **JDK**, poiché supporta anche la compilazione al volo di componenti Java.
* **Tomcat** funziona con **JRE** e non supporta tutte le funzionalità avanzate di WildFly.

**Domande:**

* Quale ambiente è necessario per WildFly e quale per Tomcat?
* Perché WildFly può essere preferibile per lo sviluppo di applicazioni Java dinamiche?

**4. JSP e Servlet**

Le **JSP** (Java Server Pages) combinano **HTML** e **Java**, mentre le **Servlet** sono classi Java che intercettano le richieste HTTP. La differenza è che:

* **JSP** è usato per il lato visivo (presentazione).
* **Servlet** è più adatto per la logica di business.

Esempio di JSP:

<%@ page language="java" contentType="text/html; charset=ISO-8859-1" pageEncoding="ISO-8859-1"%>

<!DOCTYPE html>

<html>

<head>

<meta charset="ISO-8859-1">

<title>Saluto da JSP</title>

</head>

<body>

   <%= "Ciao da JSP!" %>

</body>

</html>

​

​**Domande:**

* Qual è la differenza tra JSP e Servlet?
* Perché JSP è più adatto per la presentazione rispetto a una Servlet?

**5. Funzionamento delle Servlet**

Le Servlet contengono metodi protetti come doGet e doPost, i quali gestiscono le richieste in base al tipo (GET, POST). Questi metodi vengono chiamati dal server stesso e sono responsabili di processare la richiesta (request) e di inviare la risposta (response).

Esempio di doGet in una Servlet:

**protected** void doGet(HttpServletRequest request, HttpServletResponse response)

**throws** ServletException, IOException {

   String nome = request.getParameter("nominativo");

   response.getWriter().append("<html><body><p>Benvenuto, " + nome + "!</p></body></html>");

}

​**Domande:**

* Cosa sono request e response e come funzionano nelle Servlet?
* Come si usa getParameter per ottenere parametri dalla richiesta?

**6. Parametri e Attributi in JSP e Servlet**

Nelle Servlet, puoi usare getParameter per ottenere i dati inviati dal client (ad esempio, da un form HTML). Tuttavia, per trasmettere dati da una Servlet a una JSP, usa setAttribute e getAttribute:

* **getParameter**: ottiene un valore dalla richiesta.
* **setAttribute/getAttribute**: per passare dati tra Servlet e JSP.

Esempio di setAttribute:

request.setAttribute("utente", "Mario Rossi");

request.getRequestDispatcher("/home.jsp").forward(request, response);

​**Domande:**

* Qual è la differenza tra getParameter e getAttribute?
* Come si usa setAttribute per inviare dati a una JSP?

**7. Deployment: JAR, WAR, EAR**

* **JAR** (Java Archive): file che contiene le classi Java compilate.
* **WAR** (Web Application Archive): contiene tutte le risorse di una web app.
* **EAR** (Enterprise Archive): contenitore logico per progetti più complessi (con più WAR o JAR).

**Domande:**

* Qual è la differenza tra JAR, WAR ed EAR?
* Perché si usa un file WAR per le web application?

**8. Expression Language (EL)**

L’Expression Language in JSP è usata per accedere in modo semplice agli oggetti nelle JSP, come attributi o parametri.

Esempio di EL:

<c:forEach items="${utenti}" var="utente">

  ${utente.nome}

</c:forEach>

​**Domande:**

* A cosa serve l’Expression Language in JSP?
* Come funziona l’iterazione con <c:forEach>?

**9. HTTP Stateless e Gestione delle Sessioni**

L’HTTP è **stateless**, quindi non mantiene lo stato delle connessioni. Le sessioni sono usate per ricordare i dati dell’utente tra le richieste. Puoi implementare le sessioni per creare funzionalità come un carrello o una login persistente.

**Domande:**

* Cosa significa che HTTP è stateless?
* Come puoi mantenere lo stato di un utente in una web app?

**10. Esercizio Pratico: Creazione di una Login con JSP e Servlet**

**Obiettivo:**

Creare una login che autentica l'utente tramite localhost:8080 e WildFly.

1. **Servlet di Login** (LoginServlet.java)

**protected** void doPost(HttpServletRequest request, HttpServletResponse response)

**throws** ServletException, IOException {

   String username = request.getParameter("username");

   String password = request.getParameter("password");

**if**("utente".equals(username) && "password".equals(password)) {

       request.getSession().setAttribute("loggedUser", username);

       response.sendRedirect("home.jsp");

  } **else** {

       response.getWriter().append("Login fallito! Riprova.");

  }

}

1. ​**JSP per il Login** (login.jsp)

jsp

Copia codice

1. **JSP di Benvenuto** (home.jsp)

<html>

<head><title>Login</title></head>

<body>

   <form action="LoginServlet" method="post">

      Username: <input type="text" name="username"><br>

      Password: <input type="password" name="password"><br>

       <input type="submit" value="Login">

   </form>

</body>

</html>

​**Spiegazione:**

* **LoginServlet**: riceve i dati di login e verifica username e password.
* **login.jsp**: pagina di input per username e password.
* **home.jsp**: pagina di benvenuto per l'utente loggato.

**Domande:**

* Come passare parametri da una pagina JSP a una Servlet?
* Come si crea e gestisce una sessione utente?