

Programació

UT12.1Servlets

Introducció

- Fins ara, hem executat els nostres programes fent ús de l'entrada/eixida estàndard que ens ofereix Java (consola).
- En esta unitat aprendrem a interactuar amb el nostre codi Java des d'una aplicació web.
- Per això, Java proveeix de dos elements principals permetran esta interactuació. Per un costat els Servlets i per un altre, estretament relacionades, les pàgines JSP.

Què és unServlet?

- Un servlet (actualment Jakarta Servlet) és un fitxer Java que s'executa en un servidor web (Apache Tomcat, GlassFish, etc...)
- Esta ferramenta ens permetrà crear aplicacions web.

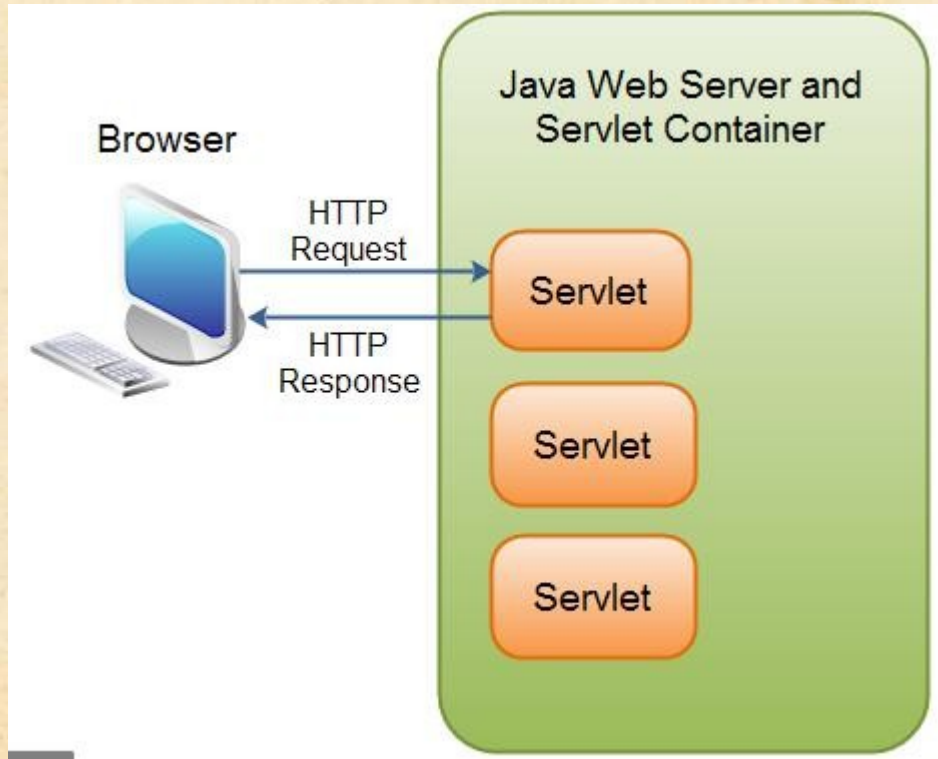


Apache
Tomcat



Web Server

En què consisteix?



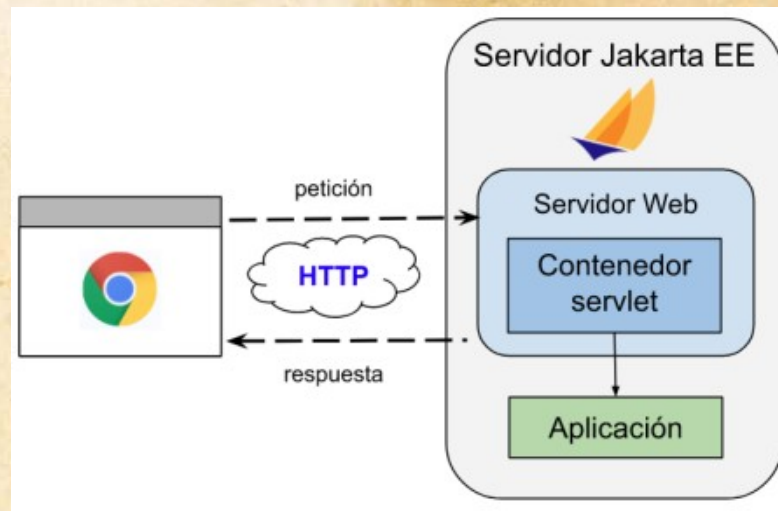
El Servlet rep una petició (request) del client (per exemple un navegador web)

El Servlet executa el seu codi java podent resoldre diferents tasques, com per exemple:

- Generar una resposta HTML, XML, PDF...
- Fer una trucada a un altre servlet
- Iniciar cookies o variables de sessió
- Fer d'enllaç entre el client i un sistema de persistència de dades (controlador)

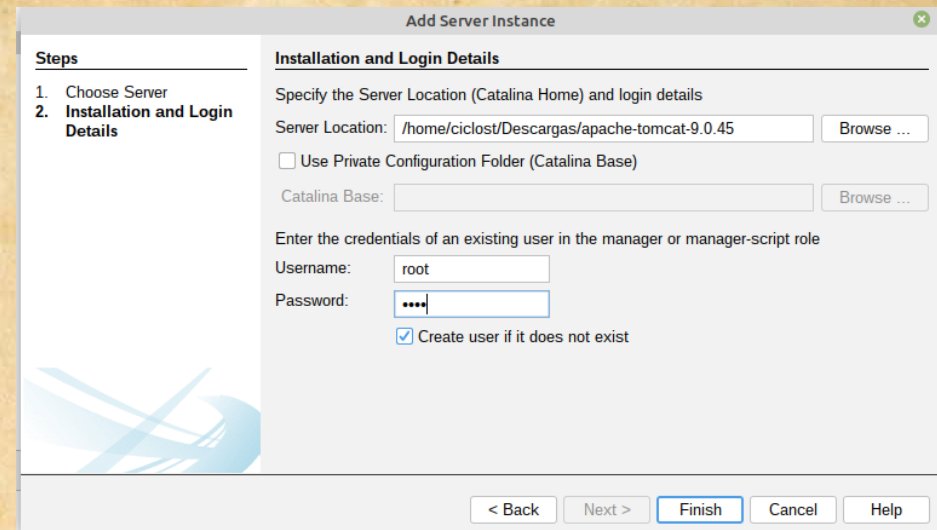
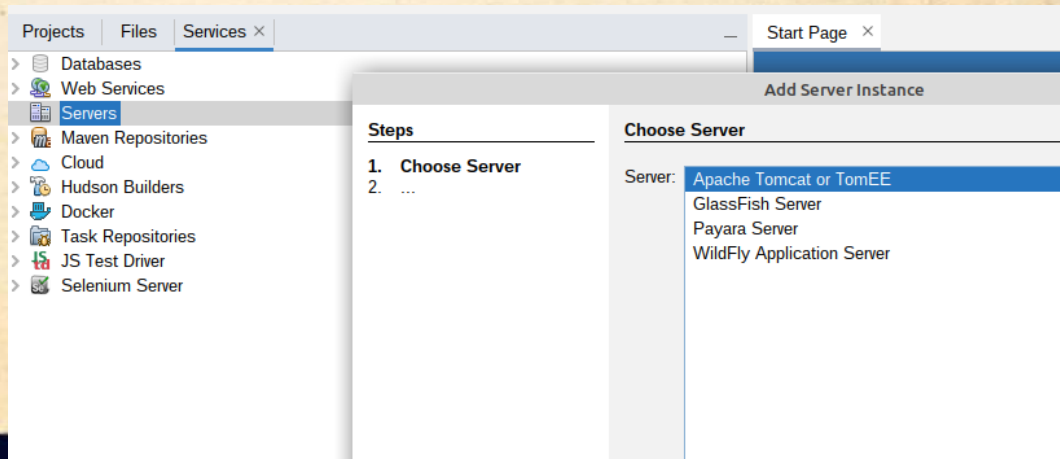
De què està compost?

- Un Servlet conté al seu interior codi Java
- Per tant, un Servlet, al igual que els nostres programes, també tindran extensió .java



Passos per a la creació d'una Aplicació Web en Netbeans

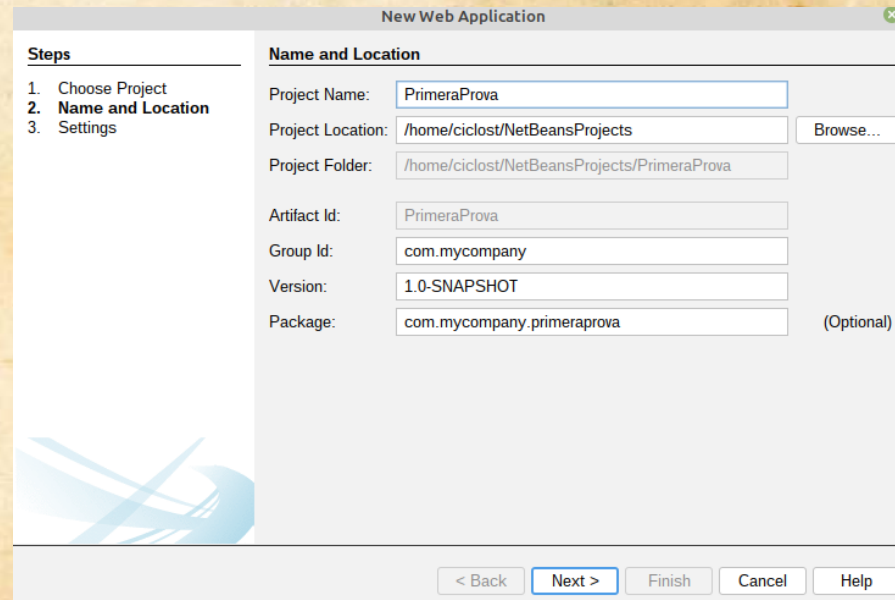
- Per a poder desenvolupar aplicacions web cal tindre un servidor web on poder desplegar-les.
- En el nostre cas descarregarem el servidor [Apache Tomcat](#) i descomprimirem la carpeta.
- Una volta descomprimida la carpeta hem de donar d'alta el servidor web en NetBeans



En Server Location, indica la carpeta arrel d'Apache Tomcat usuari "root" contrasenya "root"

Passos per a la creació d'una Aplicació Web en Netbeans

- El següent pas és crear un projecte tipus Aplicació Web en Netbeans.
- File-> New Project -> Java with Maven-> Web Application
- Donem un nom al projecte i premem Next:



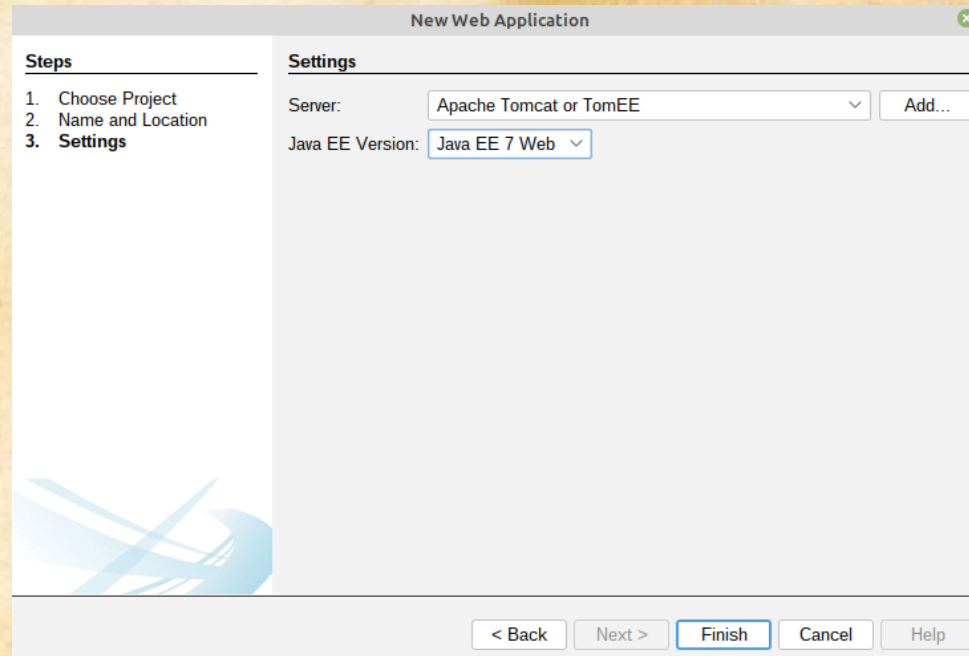
The screenshot shows the 'New Web Application' dialog box in NetBeans. The 'Steps' panel on the left indicates the current step is '2. Name and Location'. The 'Name and Location' panel contains the following fields:

Name and Location	
Project Name:	PrimeraProva
Project Location:	/home/ciclost/NetBeansProjects Browse...
Project Folder:	/home/ciclost/NetBeansProjects/PrimeraProva
Artifact Id:	PrimeraProva
Group Id:	com.mycompany
Version:	1.0-SNAPSHOT
Package:	com.mycompany.primeraProva (Optional)

At the bottom of the dialog, there are navigation buttons: '< Back', 'Next >', 'Finish', 'Cancel', and 'Help'. The 'Next >' button is highlighted.

Passos per a la creació d'una Aplicació Web en Netbeans

- El següent pas és indicar quin serà el servidor web que utilitzarà la nostra aplicació.
- Seleccionarem el servidor que hem instal·lat anteriorment.

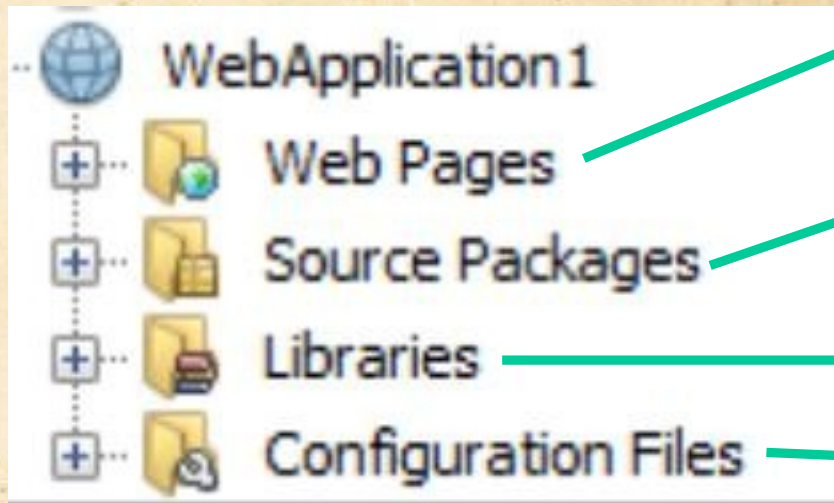
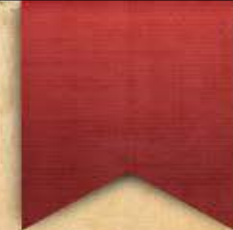


Passos per a la creació d'una Aplicació Web en Netbeans

Finalment, si volem treballar amb bases de dades afegirem les dependències Maven a `Project Files > pom.xml`

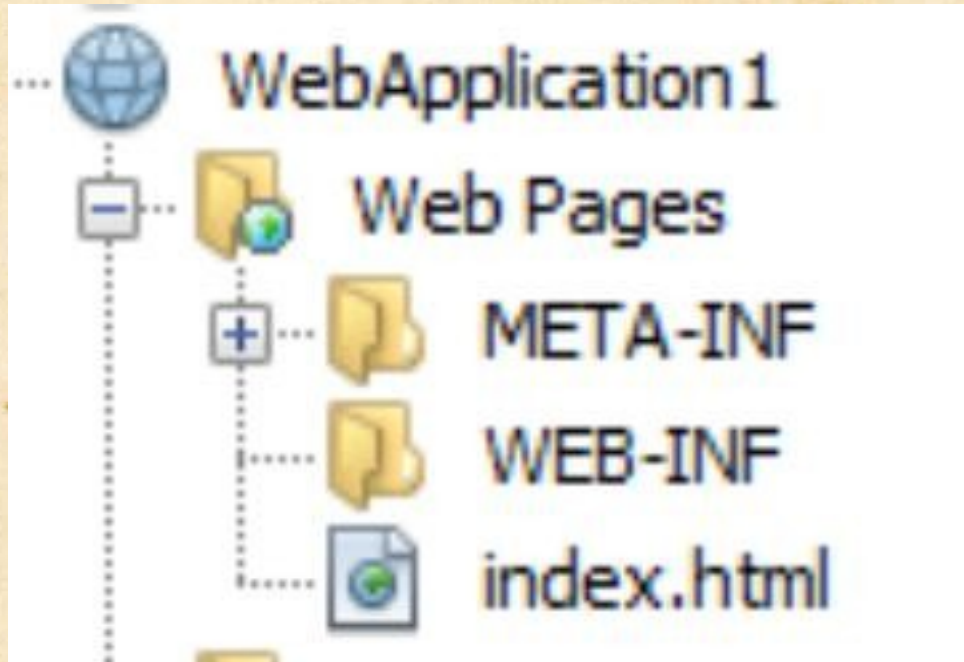
```
<dependencies>
  <dependency>
    <groupId>com.oracle.database.jdbc</groupId>
    <artifactId>ojdbc8</artifactId>
    <version>21.5.0.0</version>
  </dependency>
  <dependency>
    <groupId>org.apache.tomcat</groupId>
    <artifactId>tomcat-jdbc</artifactId>
    <version>10.1.0-M12</version>
  </dependency>
  <dependency>
    <groupId>javax</groupId>
    <artifactId>javaee-web-api</artifactId>
    <version>7.0</version>
    <scope>provided</scope>
  </dependency>
</dependencies>
```


Estructura d'una Aplicacio Web



- Carpeta on s'ubicaran tots els fitxers relacionats amb les pàgines web(html,css, javascript) i les pàgines jsp. Pot contindre subcarpetes (css, imatges...)
- Carpeta on s'ubicaran els fitxers java (servlets, llibreries pròpies, capa de model)
- Llibreries addicionals al projecte (JUnit, etc)
- Fitxers de configuració per al desplegament de l'aplicació web

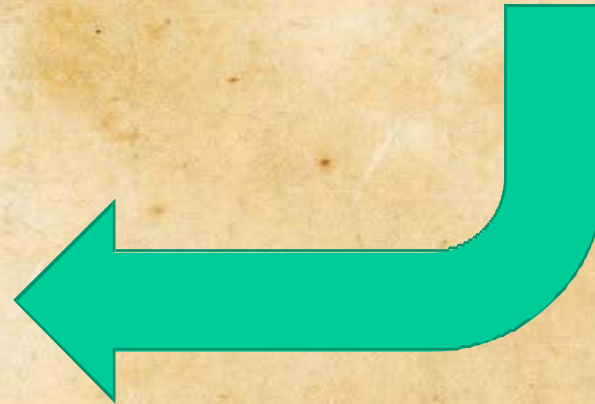
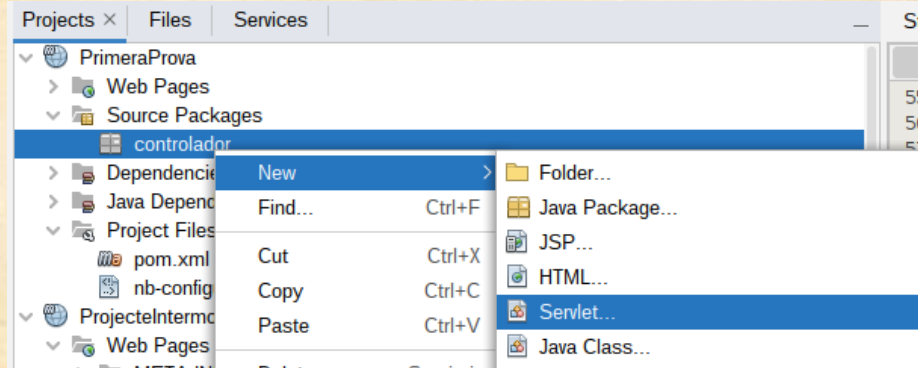
Estructura d'una Aplicació Web



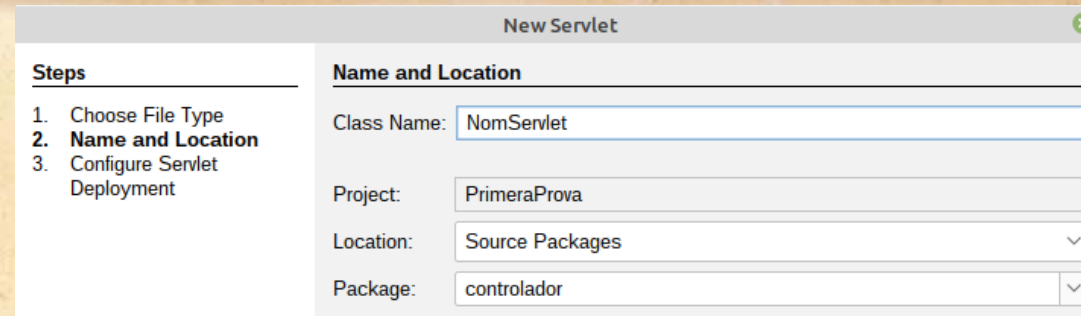
- META-INF: Emmagatzema normalment el fitxer `context.xml` que guarda **informació de configuració** (per exemple una font de dades o la ruta al projecte)
- WEB-INF: Emmagatzema normalment el fitxer `web.xml` que és un **descriptor d'implementació**. Guarda informació sobre de les classes, recursos i configuració que utilitza el servidor per a resoldre les sol·licituds web que rep.
- El fitxer `index.html` o bé `index.jsp` serà la pàgina d'inici per defecte

Creació d'un nou Servlet

Fem clic dret sobre un paquet dins de Source Packages, i tindrem l'opció de "Servlet"

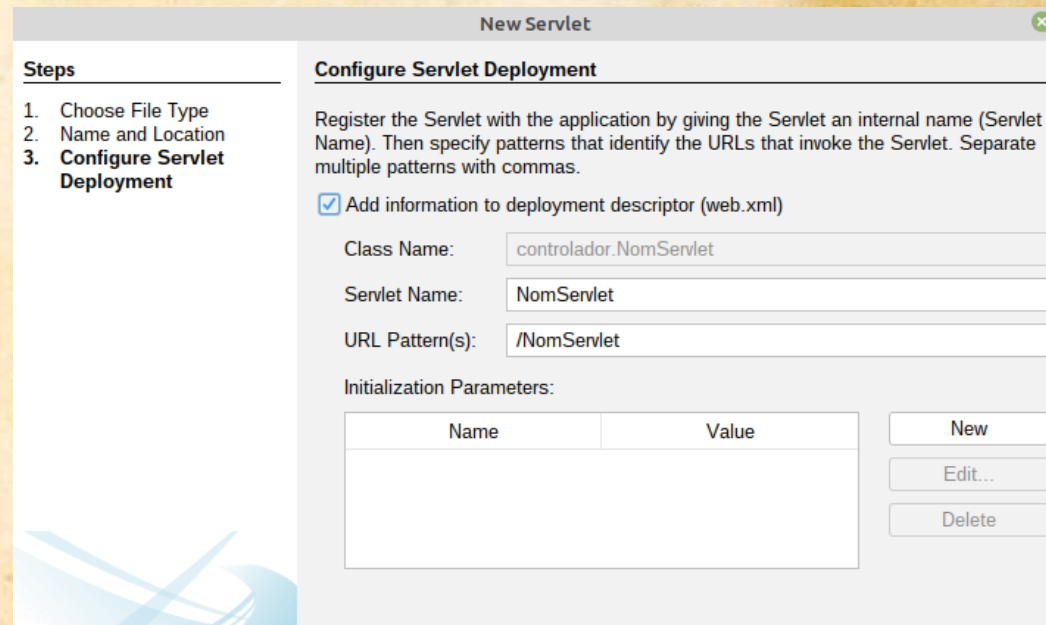


Creació d'un nou Servlet



The screenshot shows the 'New Servlet' dialog box with the 'Name and Location' tab selected. The 'Steps' list on the left indicates the current step is '2. Name and Location'. The 'Class Name' field is set to 'NomServlet'. The 'Project' dropdown is set to 'PrimeraProva'. The 'Location' dropdown is set to 'Source Packages'. The 'Package' dropdown is set to 'controlador'.

Steps	Name and Location
1. Choose File Type	Class Name: NomServlet
2. Name and Location	Project: PrimeraProva
3. Configure Servlet Deployment	Location: Source Packages
	Package: controlador

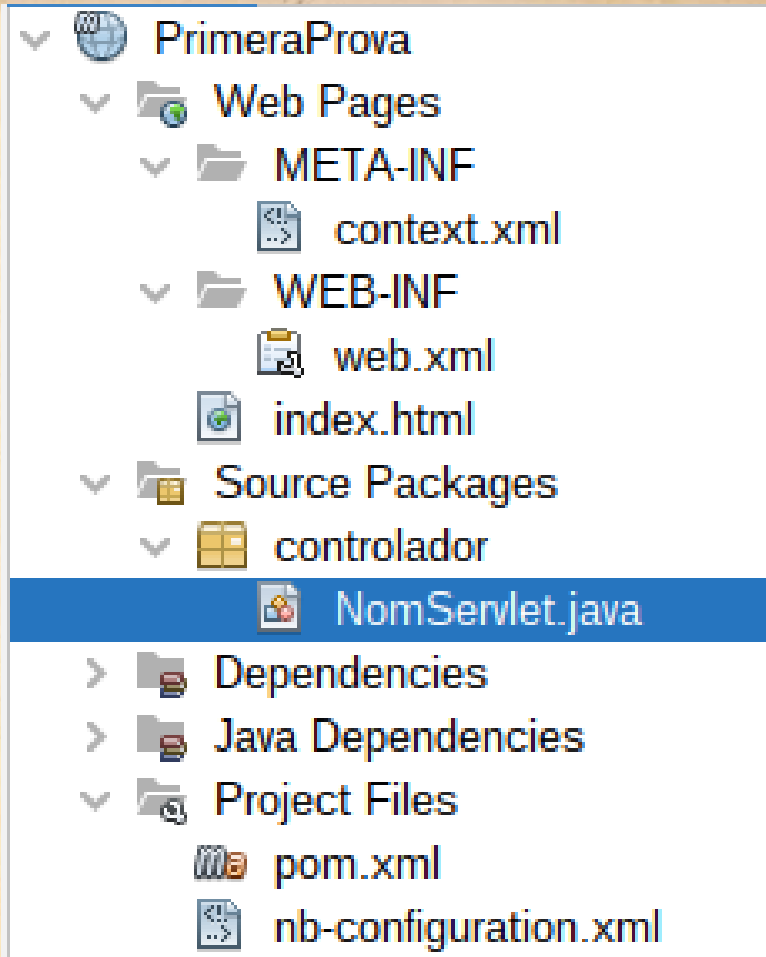


The screenshot shows the 'New Servlet' dialog box with the 'Configure Servlet Deployment' tab selected. The 'Steps' list on the left indicates the current step is '3. Configure Servlet Deployment'. The 'Add information to deployment descriptor (web.xml)' checkbox is checked. The 'Class Name' field is set to 'controlador.NomServlet'. The 'Servlet Name' field is set to 'NomServlet'. The 'URL Pattern(s)' field is set to '/NomServlet'. The 'Initialization Parameters' section is empty.

Steps	Configure Servlet Deployment		
1. Choose File Type	Register the Servlet with the application by giving the Servlet an internal name (Servlet Name). Then specify patterns that identify the URLs that invoke the Servlet. Separate multiple patterns with commas.		
2. Name and Location	<input checked="" type="checkbox"/> Add information to deployment descriptor (web.xml)		
3. Configure Servlet Deployment	Class Name: controlador.NomServlet		
	Servlet Name: NomServlet		
	URL Pattern(s): /NomServlet		
	Initialization Parameters:		
	<table border="1"><thead><tr><th>Name</th><th>Value</th></tr></thead><tbody></tbody></table>	Name	Value
Name	Value		

**Activem el
check "Add
information to
deployment
descriptor!!!**

Creació d'un nou Servlet



El Servlet apareix al paquet que hem seleccionat, sempre dins de “**Source Packatges**”

Codi bàsic de un Servlet

- Quan obrim el Servlet creat observem que:
 1. Hereta de la classe `HttpServlet` que implementa `Servlet`
 2. Sobreescriu els seus mètodes `doGet` i `doPost`
 - Eixos mètodes són els encarregats de rebre la petició HTTP (request) del client i processar la seua informació.
 - Els dos mètodes reben dos paràmetre sinterfície `HttpServletRequest` i `HttpServletResponse` que representen la petició (request) rebuda i la resposta (response) que es tornarà.
 - `HttpServletResponse` implementa a `ServletResponse` que disposa d'un mètode `getWriter()` que retorna un objecte de la classe `PrintWriter`, el qual ens permet escriure una resposta en text pla (habitualment HTML).

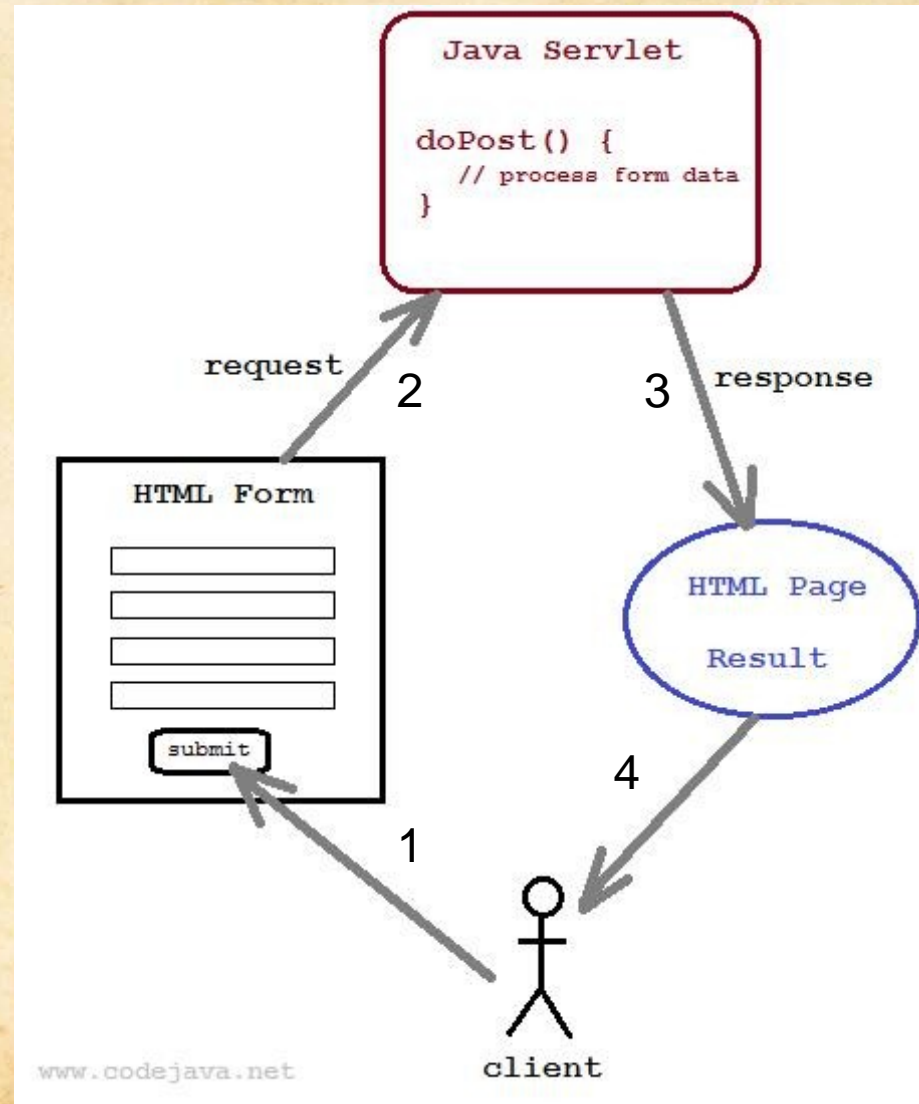
Codi bàsic de un Servlet

3. `doGet` i `doPost` invoquen a un mètode comú `processRequest`. Això vol dir que per defecte tant les peticions **get** com les **post** seran tractades al mateix lloc en el Servlet. (podriem fer tractament distinta per a get i per a post sense eixa invocació)

En `processRequest`, s'estableix el format de la resposta (per defecte `text/html`) a través del mètode `setContentType` de `response`.

I a més, s'utilitza l'objecte tornat per `getWriter()` per a escriure el codi de resposta que li arribarà al client.

Petició del Client i resposta del Servlet

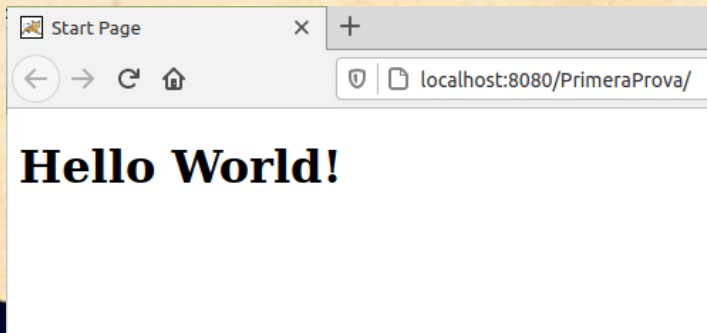


Execució d'una Aplicació Web

- Abans d'executar l'aplicació web creada per Netbeans, és convenient que faces alguns canvis a elprojecte:
 1. Decideix quin navegador s'obrirà per llançar la teua aplicació (ve definit el navegador per defecte). Per modificar-ho, prem el botó dret i modifica el navegador dins de l'opció Run > Set Project Browser
 2. Decideix si vols que la teua aplicació es desplegue de nou al servidor cada vegada que faces un canvi al codi font. Per modificar-ho, entra a les propietats del projecte, i dins de Run (opció Deploy on Save)

Execució d'una Aplicació Web

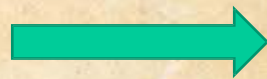
- Clica el botó dret sobre el projecte i tria la opció *Run* com és habitual amb els projectes de Java que has fet anteriorment.
- Demanarà les dades d'accés al servidor Apache Tomcat (root-root) i automàticament arrancarà el servidor web i apareixerà el navegador amb la pàgina “index.html” oberta (inici de la app)



Fixa't que podem connectar amb el nostre servidor posant a la barra de direccions: “localhost:8080”. A més, per connectar amb l'aplicació web posarem el seu nom a continuació.

Execució d'una Aplicació Web

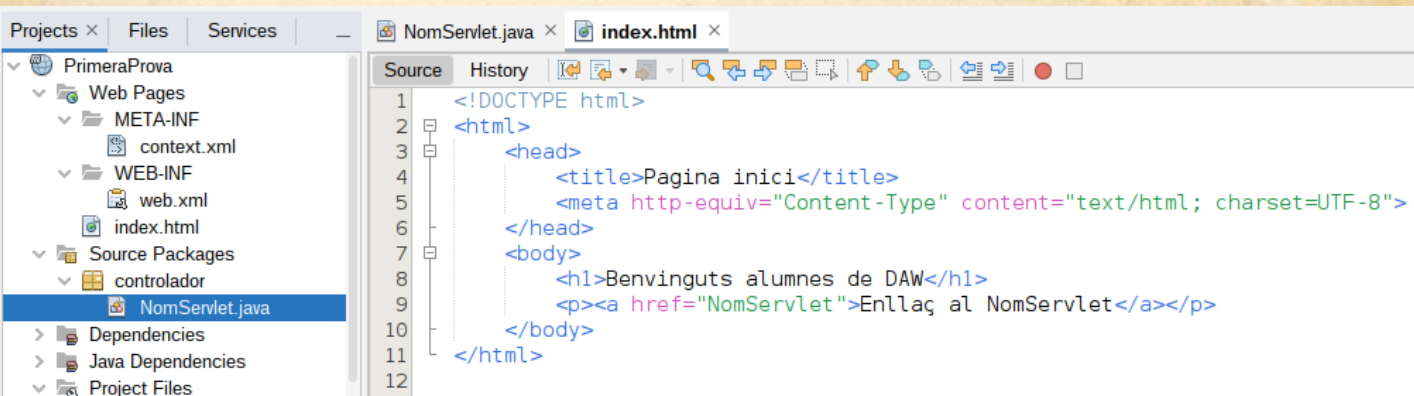
- Si obris el `fitxerindex.html` per defecte que crea Netbeans, observaràs que només fa que interpretar el codi HTML per part del navegador.
- És a dir, **no està realitzant cap petició** a cap Servlet.



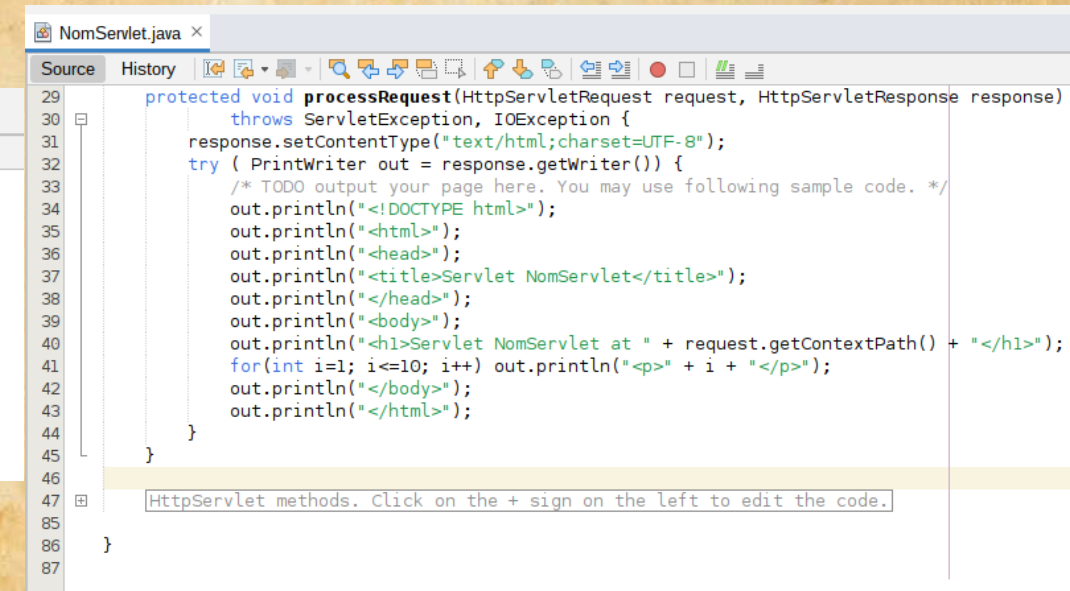
```
<!DOCTYPE html>
<html>
  <head>
    <title>Start Page</title>
    <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
  </head>
  <body>
    <h1>Hello World!</h1>
  </body>
</html>
```


Com realitzem una petició a un Servlet?

- Per connectar amb un Servlet es deu realitzar una trucada al mateix des del client.
- Una de les maneres és introduir la ruta del Servlet en l'atribut `action` d'un formulari HTML. Encara que també es podria fer amb un simple enllaç HTML

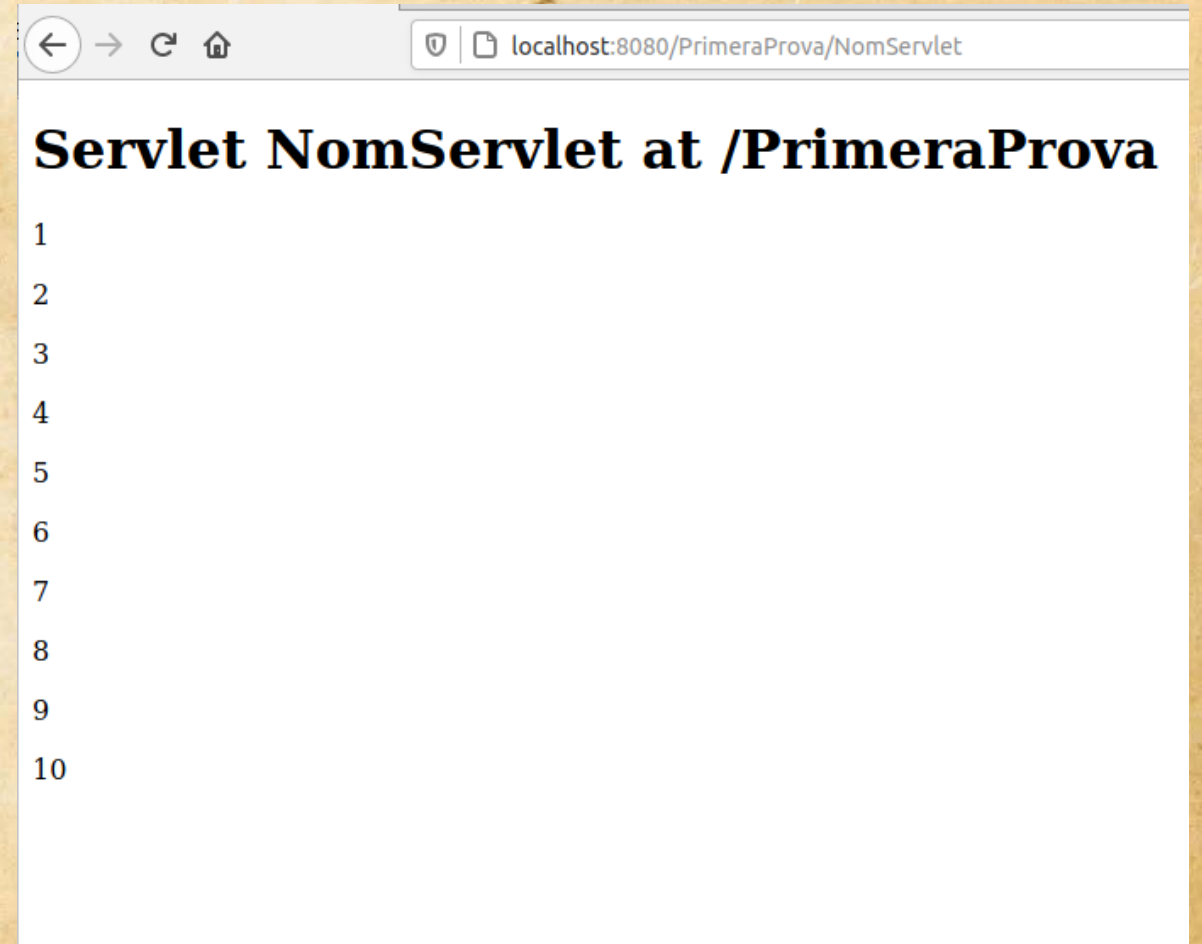
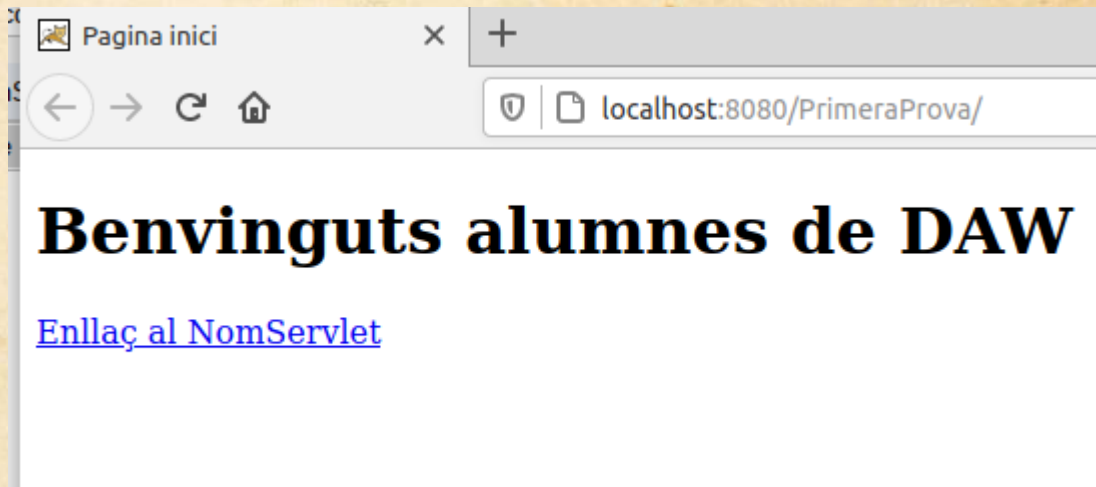


```
1 <!DOCTYPE html>
2 <html>
3   <head>
4     <title>Pagina inici</title>
5     <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
6   </head>
7   <body>
8     <h1>Benvinguts alumnes de DAW</h1>
9     <p><a href="NomServlet">Enllaç al NomServlet</a></p>
10  </body>
11 </html>
12
```



```
29 protected void processRequest(HttpServletRequest request, HttpServletResponse response)
30     throws ServletException, IOException {
31     response.setContentType("text/html;charset=UTF-8");
32     try (PrintWriter out = response.getWriter()) {
33         /* TODO output your page here. You may use following sample code. */
34         out.println("<!DOCTYPE html>");
35         out.println("<html>");
36         out.println("<head>");
37         out.println("<title>Servlet NomServlet</title>");
38         out.println("</head>");
39         out.println("<body>");
40         out.println("<h1>Servlet NomServlet at " + request.getContextPath() + "</h1>");
41         for(int i=1; i<=10; i++) out.println("<p> + i + "</p>");
42         out.println("</body>");
43         out.println("</html>");
44     }
45 }
46
47 HttpServlet methods. Click on the + sign on the left to edit the code.
48
85
86
87 }
```


Com realitzem una petició a un Servlet?



Exemple de petició a un Servlet des del client.

- Descàrrega el següent fitxer `index.html` i substitueix-lo per el que s'ha creat per defecte:

<https://aules.edu.gva.es/fp/mod/resource/view.php?id=1602646>

- Crea un nou Servlet anomenat `PrimerServlet`

Exemple de petició a un Servlet des del client

```
<form id="registro" method="post" action="PrimerServlet" >
  <h2>Emplena el següent formulari de registre</h2>
  <label for="nom">Nom: </label><br>
  <input type="text" id="nom" name="nom" size="30" maxlength="30" required autofocus /><br>
  <label for="cognoms">Cognoms</label><br>
  <input type="text" id="cognoms" name="cognoms" size="80" maxlength="80" required /><br>
  <label for="email">Email</label><br>
  <input type="email" id="email" name="email" required /><br>
  <input type="submit" name="enviar" value="Guardar canvis" />
  <input type="reset" name="reset" value="Esborrar" />
</form>
```

- En la propietat `action` del formulari escrivim el nom del Servlet on volem enviar la informació. Com les dades s'envien amb el mètode “**post**”, s'invocarà al mètode “`doPost`” de `PrimerServlet`

Exemple de petició a un Servlet des del client

- Si executem el projecte observarem que el navegador mostra el formulari que hem creat en `index.html`.
- Si fem clic en “Guardar canvis” obtindrem la següent eixida.

Exemple Primer Servlet

Emplena el següent formulari de registre

Nom:

Johny

Cognoms

Hosa Bia

Email

jonnyhb@iespacomolla.es

Guardar canvis

Esborrar



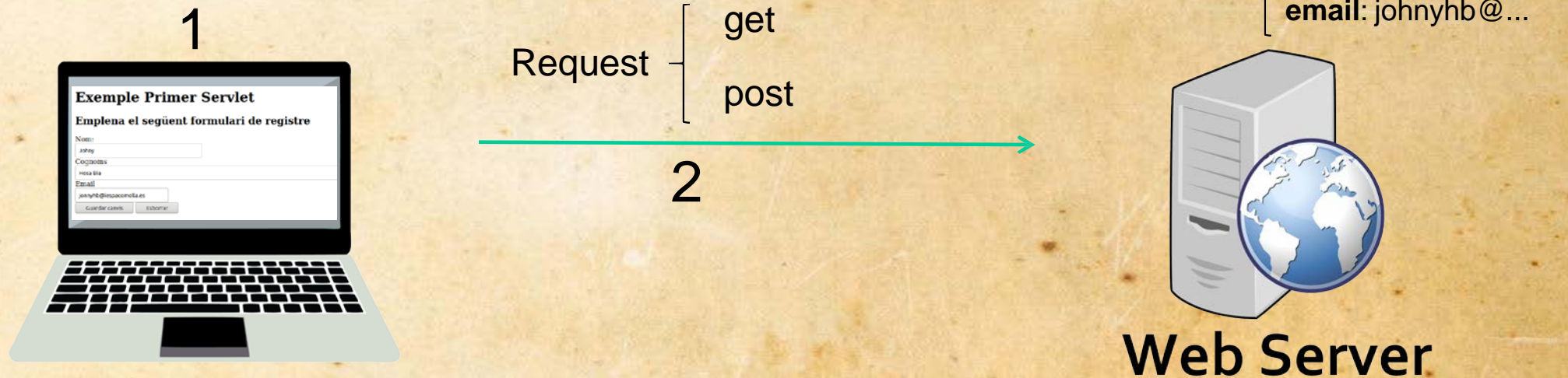
Exemple de petició a un Servlet des del client

- Si obris el codi de PrimerServlet, podràs comprovar que està mostrant el codi HTML que ha generat el mètode processRequest

```
protected void processRequest(HttpServletRequest request, HttpServletResponse response)
    throws ServletException, IOException {
    response.setContentType("text/html;charset=UTF-8");
    try (PrintWriter out = response.getWriter()) {
        /* TODO output your page here. You may use following sample code. */
        out.println("<!DOCTYPE html>");
        out.println("<html>");
        out.println("<head>");
        out.println("<title>Servlet PrimerServlet</title>");
        out.println("</head>");
        out.println("<body>");
        out.println("<h1>Servlet PrimerServlet at " + request.getContextPath() + "</h1>");
        out.println("</body>");
        out.println("</html>");
    }
}
```


Captura d'informació proporcionada des de la petició

- A el exemple anterior hem vist com es pot establir comunicació amb un Servlet.
- A la gran majoria de casos, el Servlet necessitarà de la informació que rep del client:



Captura d'informació proporcionada des de la petició

- Amb la informació rebuda el servidor podrà realitzar diferents accions: emmagatzemar-la en una BD, processar-la i donar una resposta al client, etc...
- Per a donar una resposta al client farà ús de response. Anem a veure com capturar la informació que envia el client.

Captura d'informació proporcionada des de la petició

- La variable que conté tota la informació enviada és `HttpServletRequest request`.
- Farem ús del mètode `getParameter` per a rebre la informació enviada.
- [https://docs.oracle.com/javaee/6/api/javax/servlet/ServletRequest.html#getParameter\(java.lang.String\)](https://docs.oracle.com/javaee/6/api/javax/servlet/ServletRequest.html#getParameter(java.lang.String))
- Este mètode rep el nom (name) del component del formulari i retorna la informació **en format String**

Exemple pel formulari anterior

```
protected void processRequest(HttpServletRequest request, HttpServletResponse response)
    throws ServletException, IOException {
    response.setContentType("text/html;charset=UTF-8");
    try ( PrintWriter out = response.getWriter()) {
        /* TODO output your page here. You may use following sample code. */
        out.println("<!DOCTYPE html>");
        out.println("<html>");
        out.println("<head>");
        out.println("<title>Formulari d'exemple rebut</title>");
        out.println("</head>");
        out.println("<body>");
        out.println("<h1>Formulari " + request.getContextPath() + " rebut! </h1>");
        out.println("<p><strong>Nom:</strong> " + request.getParameter("nom") + " - rebut</p>");
        out.println("<p><strong>Cognoms:</strong> " + request.getParameter("cognoms") + " - rebut</p>");
        out.println("<p><strong>Email:</strong> " + request.getParameter("email") + " - rebut</p>");
        out.println("</body>");
        out.println("</html>");
    }
}
```

```
<form id="registro" method="post" action="PrimerServlet">
  <h2>Emplena el següent formulari de registre</h2>
  <label for="nom">Nom: </label><br>
  <input type="text" id="nom" name="nom" size="30" maxlength="30" required autofocus /><br>
  <label for="cognoms">Cognoms</label><br>
  <input type="text" id="cognoms" name="cognoms" size="80" maxlength="80" required /><br>
  <label for="email">Email</label><br>
  <input type="email" id="email" name="email" required /><br>
  <input type="submit" name="enviar" value="Guardar canvis" />
  <input type="reset" name="reset" value="Esborrar" />
</form>
```



localhost:8080/PrimeraProva/PrimerServlet

Formulari /PrimeraProva rebut!

Nom: Johny - rebut

Cognoms: Hosa Bia - rebut

Email: jonnyhb@iespacomolla.es - rebut