Servlet/JSP

Sở Thông Tin Truyền Thông Vĩnh Long
12/2012

Trình bày: TS. NGÔ BÁ HÙNG
Website: http://sites.google.com/site/nbhung

Nội dung

- Giới thiệu về Java Enterprise Edition Platform
- Công nghệ Servlet
- Công nghệ JSP

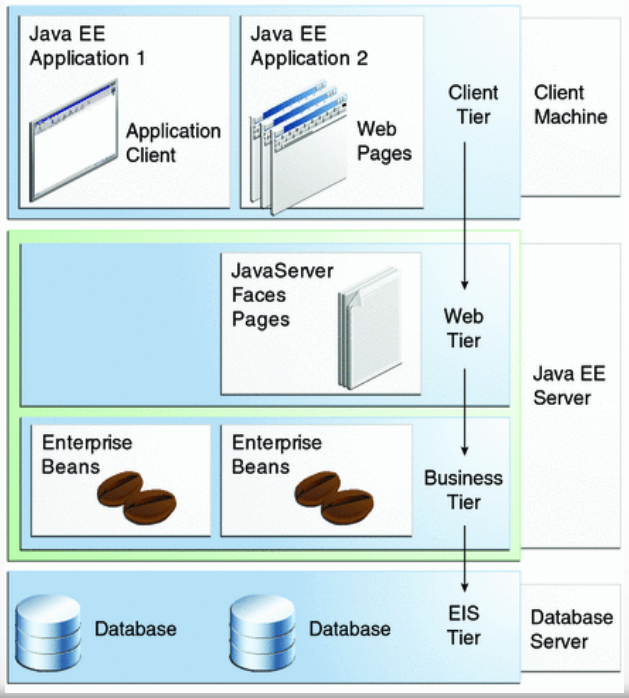
Java Platform

- Java Platform là một môi trường đặc biệt để thực thi các ứng dụng được viết bằng ngôn ngữ Java, bao gồm một Java Virtual Machine (VM) và một application programming interface (API).
- Hiện tại có 4 Programming Java Platforms
  - Java Standard Edition (Java SE) Platform
  - **Java Enterprise Edition (Java EE) Platform**
  - Java Micro Edition (Java ME) Platform
  - JavaFX

Java Enterprise Edition Platform

- Được thiết kế để giúp các nhà phát triển tạo ra các Ứng dụng doanh nghiệp (Enterprise applications) được định nghĩa là large-scale, multi-tiered, scalable, reliable, and secure network applications.
- Bao gồm một mô hình phát triển ứng dụng, một API, một môi trường thực thi để giảm độ phức tạp khi phát triển các ứng dụng doanh nghiệp, giúp các nhà phát triển tập trung vào cài đặt các chức năng của ứng dụng

Mô hình ứng dụng đa tầng

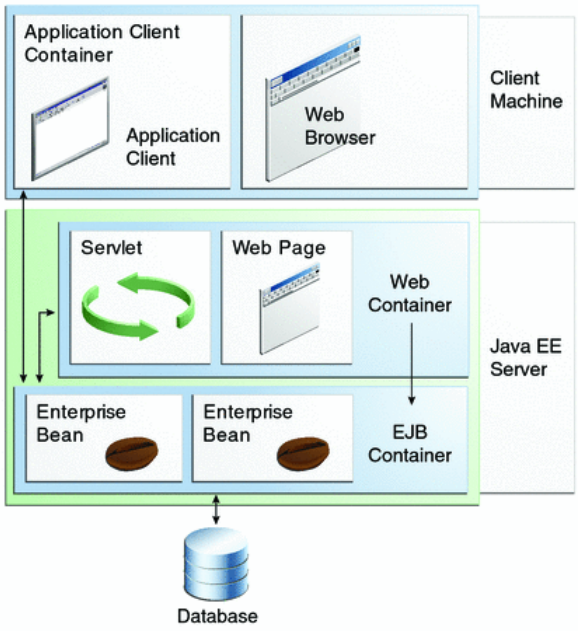


- Được sử dụng bởi Java EE Platform
- Gồm có các tầng sau:
  - Client Tier
  - Web Tier
  - Business Tier
  - Enterprise Information System
- Các tầng thực thi trên một máy tính riêng

Java EE Components/Containers

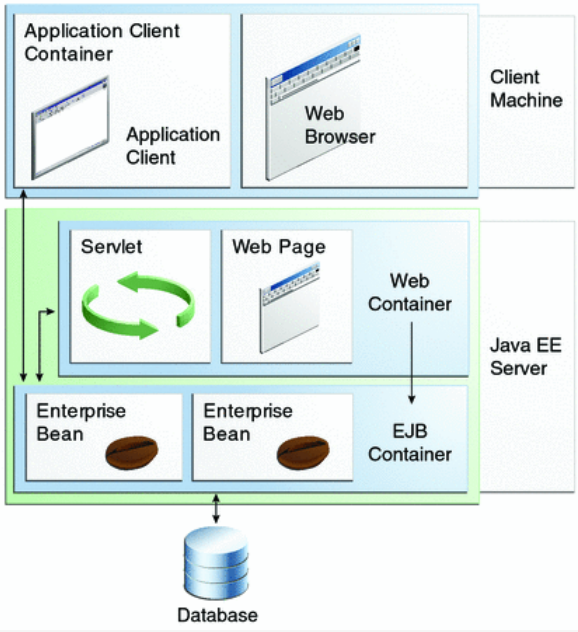
- Component là một phần mềm có tính tự chứa đựng được phát triển bằng ngôn ngữ Java
- Containers là một lớp phần mềm nền đảm bảo việc giao tiếp giữa một component với phần cứng phía dưới, là môi trường để thực thi các component và đảm bảo sự giao tiếp giữa các component với nhau
- Một ứng dụng Java EE là một tập các thành phần (components) phân bố trên nhiều Containers khác nhau

Java EE Server and Containers



- Java EE server: cung cấp Enterprise JavaBeans (EJB) container và Web container

Java EE Server and Containers

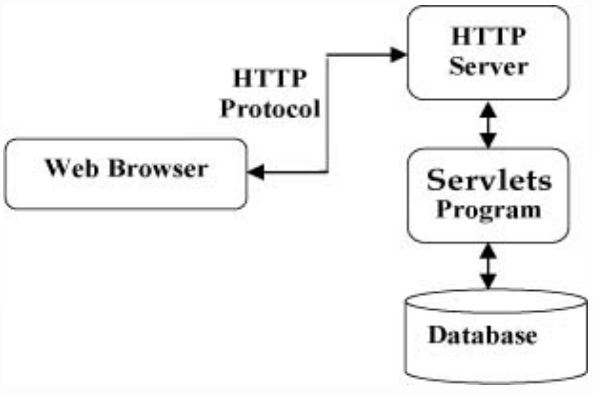


- Enterprise JavaBeans (EJB) container: quản lý sự thực thi của các enterprise beans
- Web container: quản lý sự thực thi của web pages, servlets, và một vài EJB components

Servlet

- Chương trình Java chạy trên Webserver
- Tương tác với các cơ sở dữ liệu và các chương trình khác để tạo ra các trang web với nội dung thay đổi (động) tùy thuộc tham số của yêu cầu gởi từ HTTP Client
- Có cùng mục đích cơ chế với CGI (Common Gateway Interface)
- Viết hoàn toàn bằng Java, độc lập nền
- Có thể sử dụng toàn bộ thư viện của Java

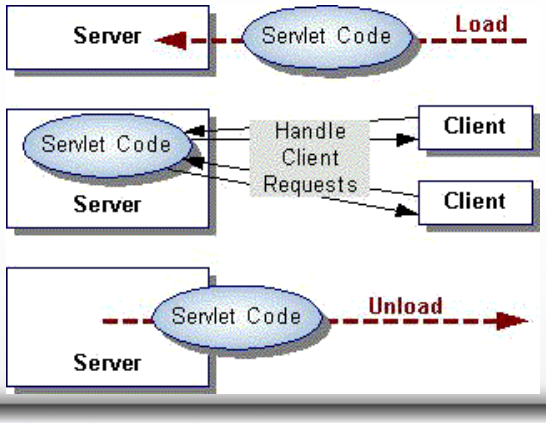
Kiến trúc/vai trò của Servlet



- Nhận yêu cầu, dữ liệu gởi đến từ HTTP Client (Browser) : Form, Cookies, media files
- Xử lý yêu cầu, dữ liệu để tạo kết quả - có thể phải truy cập cơ sở dữ liệu hoặc gọi các ứng dụng khác (RMI, Web service,...)
- Gởi dữ liệu kết quả về HTTP Client dưới dạng text (HTML, XML) hoặc nhị phân (GIF files), Excel, cookies, ...

Các thành phần của một Servlet

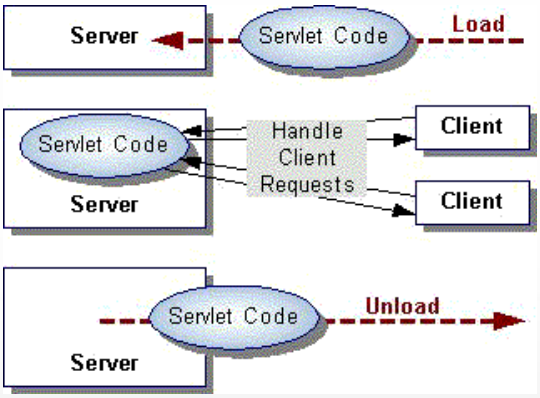
- init() – phương thức được gọi khi servlet được khởi tạo bởi Server. Thường được gọi khi phương thức doGet() hoặc doPut() của servlet được gọi lần đầu tiên
- destroy() – phương thức được gọi khi servlet bị giải phóng, thường khi tiến trình server bị dừng



http://java.sun.com/docs/books/tutorial/servlets/lifecycle/index.html

Các thành phần của một Servlet

- doGet() – phương thức được gọi khi server được truy cập đến bởi phương thức HTTP GET.
- doPost() – phương thức được gọi khi server được truy cập đến bởi phương thức HTTP POST.



http://java.sun.com/docs/books/tutorial/servlets/lifecycle/index.html

Java Server Pages (JSP)

- Servlet dùng Java để in ra mã HTML
  - Khó đọc, mất nhiều công sức nếu trang chứa nhiều văn bản
- JSP (Java Server Pages) là một cách để tạo ra các servlets
  - Viết như một trang HTML, cho phép trộn các mã Java để sinh nội dung động tùy theo yêu cầu của Client
  - Mã Java bao bọc trong thẻ đặc biệt <% …. %>
- Tập tin JSP có phần mở rộng là .jsp
  - JSP được dịch thành một servlet khi được biên dịch để chạy như các servlets đuowc viết thông thường
  - Browser/client chỉ nhận được mã HTML

Java EE Servers

Tomcat

Oracle GlassFish Server 3.x

Tested Configuration

Caucho Resin 4.0.17

Tested Configuration

JBoss Application Server 7

Tested Configuration

Apache TomEE 1.0.0-beta-1

Tested Configuration

Apache Geronimo 3.0-beta-1

Tested Configuration

Oracle GlassFish Server 3.x

Tested Configuration

IBM WebSphere Application Server 8.0

Tested Configuration

IBM WebSphere Application Server Community Edition 3.0

Tested Configuration

Oracle WebLogic Server

TMAX JEUS 7

Tested Configuration

Fujitsu Interstage Application Server powered by Windows Azure

Tested Configuration

Apache Geronimo 3.0-beta-1

Tested Configuration

Lập trình Servlet

- Cài đặt gói hỗ trợ Servlet: javax.servlet và javax.servlet.http
- Cài đặt Web server có tích hợp trình thông dịch hỗ trợ đặc tả servlet để chạy servlet
- Đặt đường dẫn CLASSPATH đến gói hỗ trợ Servlet
- Biên soạn và biên dịch Servlet như các lớp Java
  - Cài đặt các phương thức init(), doGet(), doPost(), ...
- Đưa servlet vào thư mục qui ước trên web server

Ví dụ servlet

```
import java.io.*;
import javax.servlet.*;
import javax.servlet.http.*;

public class Hellox extends HttpServlet {

public void doGet(HttpServletRequest request,
                  HttpServletResponse response)
            throws IOException, ServletException
{
    response.setContentType("text/html");
    PrintWriter out = response.getWriter();
    out.println("<html>");
    out.println("<head>");
    out.println("<title>Hello World!</title>");
    out.println("</head>");
    out.println("<body>");
    out.println("<h1>Hello World!</h1>");
    out.println("</body>");
    out.println("</html>");
} // doGet
} // Hellox
```

Tomcat

- Open Source Server hỗ trợ Servlet và JSP
- Download từ http://tomcat.apache.org
- Cài đặt và đặt các biến môi trường
  - CATALINA_HOME=Đường dẫn đến thư mục tomcat
  - Bổ sung $CATALINA_HOME/common/lib/servlet-api.jar vào CLASSPATH
- Vào thư mục $CATALINA_HOME/bin để
  - Khởi động: ./startup.sh / startup.bat
  - Tắt: ./shutdown.sh / shutdown.bat

Start tomcat

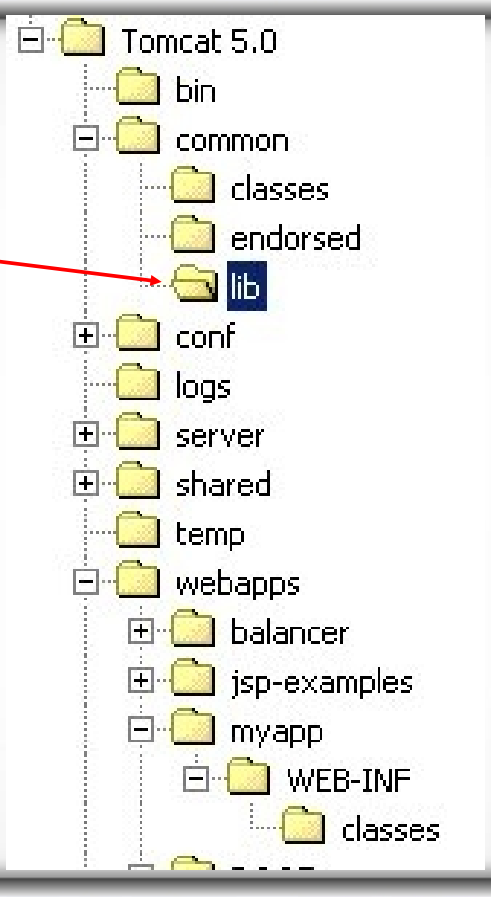
- Chuyển vào thư mục $CATALINA/bin
- Chạy không phải kiểu dịch vụ để kiểm tra
  - ./catalina.sh run catalina.bat run
- Chạy theo kiểu dịch vụ
  - ./startup.sh startup.bat
- Lưu ý: Trên linux người start phải có quyền write trên thư mục tomcat và các thư mục con

Compiling

javac –classpath

$LIB/servlet-api.jar

Hellox.java



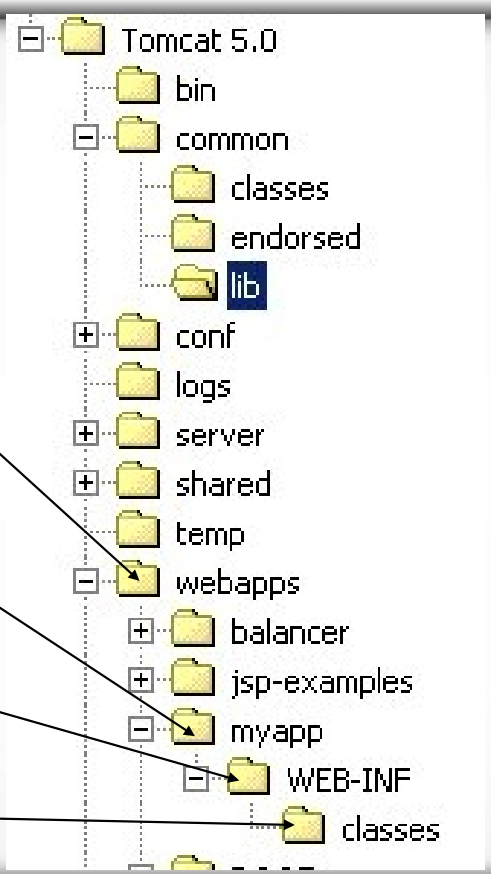
Directory Structure

Thư mục chứa các ứng dụng

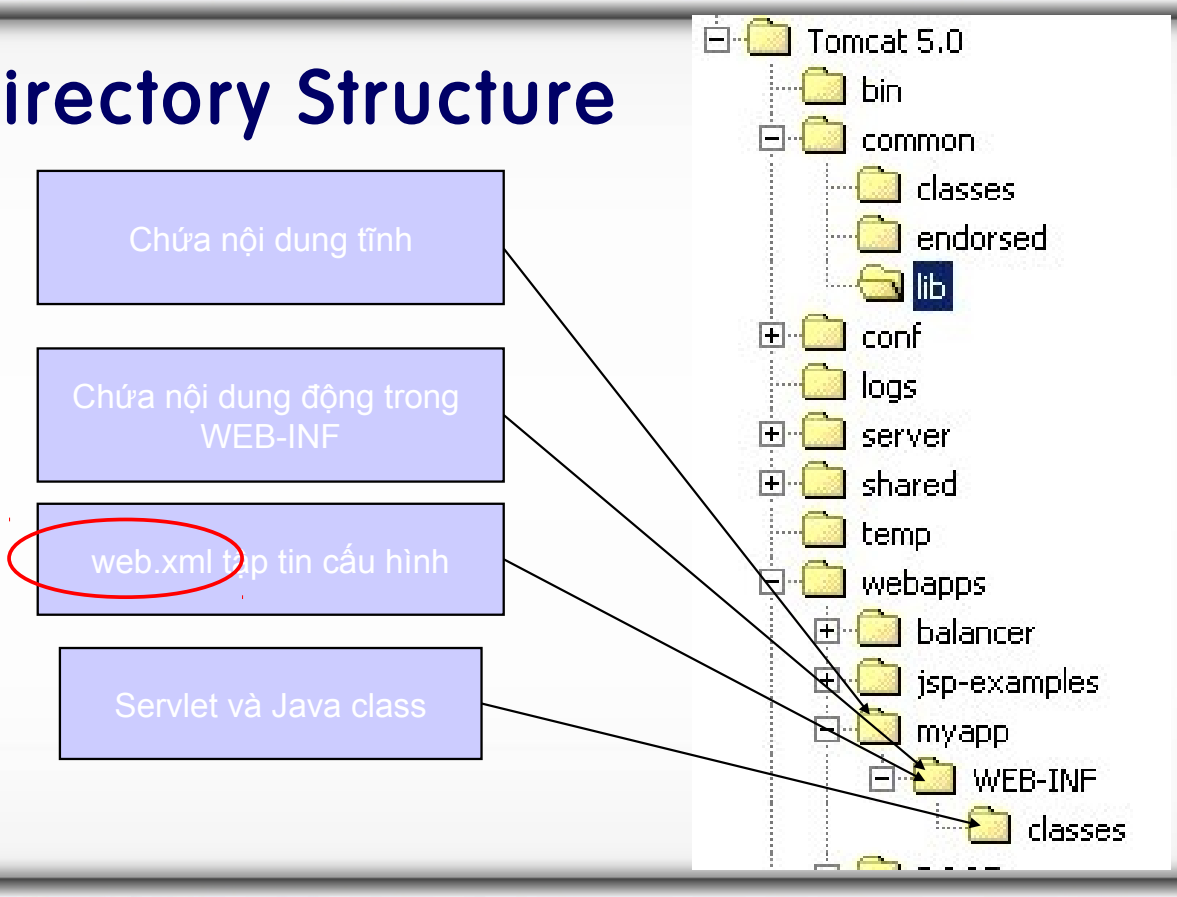
Mỗi ứng dụng có một thư mục riêng

"WEB-INF" thư mục bắt buộc

Thư mục bắt buộc chứa các lớp Java/Servlet



Directory Structure




```
<?xml version="1.0" encoding="ISO-8859-1"?>

<web-app xmlns="http://java.sun.com/xml/ns/j2ee"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation=
"http://java.sun.com/xml/ns/j2ee http://java.sun.com/xml/ns/j2ee/web-app_2_4.xsd"
    version="2.4">

    <description>Examples</description>
    <display-name>Examples</display-name>

    <servlet>
      <servlet-name>Hellox</servlet-name>
      <servlet-class>Hellox</servlet-class>
    </servlet>

    <servlet-mapping>
        <servlet-name>Hellox</servlet-name>
        <url-pattern>/Hellox</url-pattern>
    </servlet-mapping> </web-app>
```

Declares servlet

abbreviation

fully qualified (e.g., java.lang.String)

Maps servlet to URL (rooted at D)

Thực hành

- Cài đặt các phần mềm cần thiết
- Quản trị Tomcat
- Phát triển Servlet thủ công
- Phát triển Servlet sử dụng Eclipse

  [Tham khảo Slides 5-Tomcat-Eclipse]

Giới thiệu về form

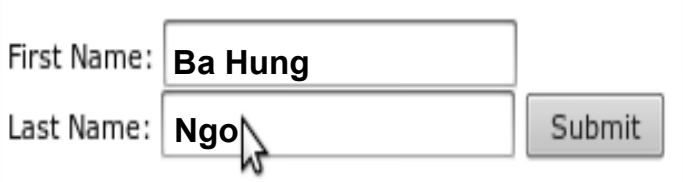
- Giao diện nhận dữ liệu nhập từ web client
  - Ví dụ: Username / Password
- Chuyển dữ liệu lên web server
- Servlet được kích hoạt bởi form sẽ
  - Nhận dữ liệu
  - Xử lý dữ liệu
  - Tạo trang web kết quả trả về cho web client

GET và POST

- Là 2 phương thức để web client chuyển dữ liệu lên web server
- GET: Dữ liệu được gởi thông quan QUERY_STRING
  - http://www.test.com/hello?key1=value1&key2=value2
- POST: Dữ liệu gởi thông qua thiết bị nhập chuẩn của chương trình

Ví dụ về phương thức GET

- http://yourserver/hello.html



- Kết quả trả về:
  - First Name: Ba Hung
  - Last Name: Ngo

hello.html

```
<html>
<body>
<form action="HelloForm" method="GET">
First Name: <input type="text" name="first_name">
<br />
Last Name: <input type="text" name="last_name" />
<input type="submit" value="Submit" />
</form>
</body>
</html>
```

http://localhost:8080/HelloForm?first_name=BH&last_name=Ngo

Servlet xử lý Form/GET

- // import các lớp thư viện cần thiết
  ```
  import java.io.*;
  import javax.servlet.*;
  import javax.servlet.http.*;
  ```
- // Định nghĩa một servlet để xử lý yêu cầu dạng GET
  ```
  public class HelloForm extends HttpServlet {
    public void doGet(HttpServletRequest request,
                      HttpServletResponse response)
              throws ServletException, IOException  {
           // ...
       // Cài đặt xử lý dữ liệu và tạo kết quả trả về ở đây
    }
  }
  ```

Cài đặt doGet()

```
response.setContentType("text/html");
PrintWriter out = response.getWriter();
String title = "Using GET Method to Read Form Data";
String docType =   "<!doctype html public \"-//w3c//dtd html 4.0 "
                                                + "transitional//en\">\n";
out.println(docType +
            "<html>\n" + "<head><title>" + title + "</title></head>\n" +
            "<body bgcolor=\"#f0f0f0\">\n" +
            "<h1 align=\"center\">" + title + "</h1>\n" +
            "<ul>\n" +
            "  <li><b>First Name</b>: "
            + request.getParameter("first_name") + "\n" +
            "  <li><b>Last Name</b>: "
            + request.getParameter("last_name") + "\n" +
            "</ul>\n" +
            "</body></html>");
```


Thực hành -helloform

- Tạo dự án có tên là helloform
- Nhấp chuột phải lên WebContent → New → HTML File → Đặt tên là hello.html
- Bổ sung nội dung hello.html như sau:
  - <form action="**/helloform/HelloForm**" method="**GET**">
  - First Name: <input type="text" name="first_name">
  - Last Name: <input type="text" name="last_name" />
  - <input type="submit" value="Submit" />
  - </form>

Thực hành -helloform

- Tạo servlet có tên là HelloForm, cài đặt phương thức doGet() cho servlet này (như ví dụ trước)
- Chèn vào web.xml

```
<servlet>
        <servlet-name>HelloForm</servlet-name>
        <servlet-class>HelloForm</servlet-class>  </servlet>
<servlet-mapping>
    <servlet-name>HelloForm</servlet-name>
    <url-pattern>/HelloForm</url-pattern>
</servlet-mapping>
```


Thực hành -helloform

- Bấm nút Stop Tomcat Server/Eclipse
- Nhấp chuột phải lên tên dự án helloform → Run As ->Run on Server để triển khai cả dự án lên Tomcat
- Truy cập http://localhost:8080/helloform/hello.html

Servlet xử lý Form/POST

- Bổ sung vào phương thức doPost của HelloForm:
  - public void doPost(HttpServletRequest request,
    HttpServletResponse response)
    throws ServletException, IOException {
    // Cài đặt xử lý dữ liệu và tạo kết quả trả về ở đây
    doGet(request, response);
    }
- Tạo tập tin hello2.html giống như hello.html, chỉ khác ở phương thức gọi Servlet là post:
  - <form action="/helloformHelloForm" **method="POST"**>

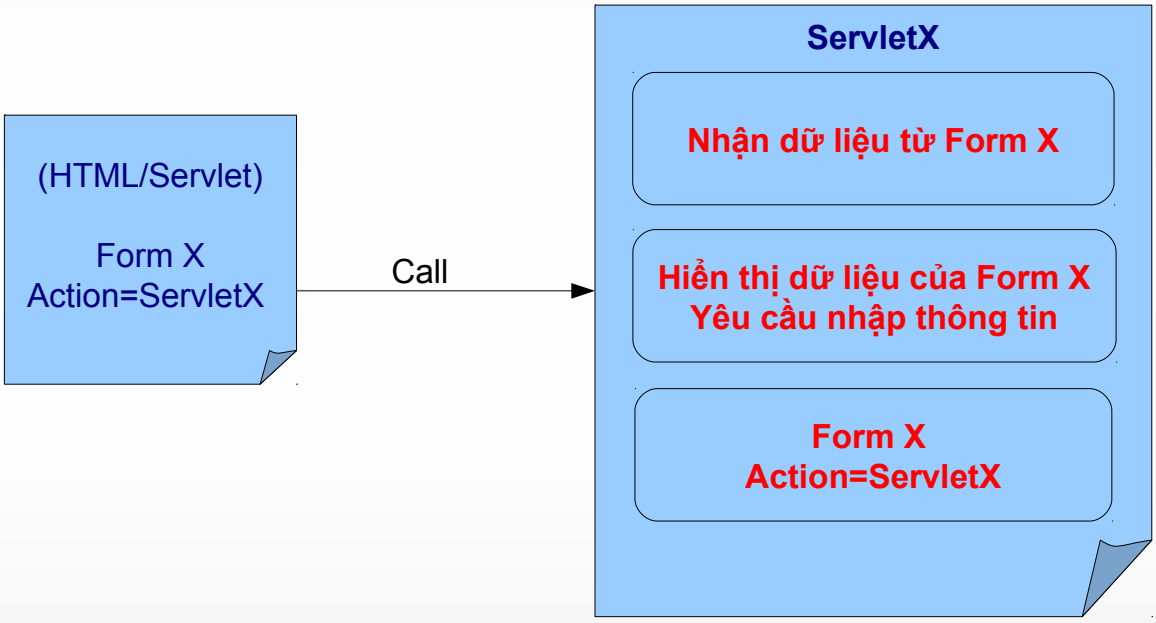
Servlet xử lý Form/POST (tt)

- Nhấp chuột phải lên tập tin hello2.html → Run As → Run on Server để triển khai tập tin hello2.html lên ứng dụng → Chấp lời đề nghị restart Tomcat Server
- Truy cập http://localhost:8080/helloform/**hello2**.html

Bài tập 1

- Viết ứng dụng web cho phép
  - Nhập vào username, password
- Nếu username=admin, password=admin
  - Trả về: Đăng nhập thành công
- Ngược lại
  - Trả về: Đăng nhập thất bại

Cấu trúc phổ biến của Servlet



```
public class Helloy extends HttpServlet {
    public void doGet(HttpServletRequest request,
                      HttpServletResponse response)
                throws IOException, ServletException {
      response.setContentType("text/html");
      PrintWriter out = response.getWriter();
      out.println("<html>");
      out.println("<body>");
      out.println("<head>");
      out.println("<title>Hello, Tell me your name!</title>");
      out.println("</head>");
      out.println("<body>");

      out.println("<h1>Hello, Tell me your name!</h1> <br>");
      out.print("<form action=\"");
      out.println("NamedHello\" method=POST>");
      out.println("<input type=text length=20 name=yourname><br>");
      out.println("<input type=submit></form>");

      out.println("</body>");
      out.println("</html>");
    }
}
```

```
public class NamedHello extends HttpServlet {
public void doPost(HttpServletRequest request,
                   HttpServletResponse response)
            throws IOException, ServletException {
  response.setContentType("text/html");
  PrintWriter out = response.getWriter();

  String name = request.getParameter("yourname");

  out.println("<html>");
  out.println("<body>");
  out.println("<head>");
  out.println("<title>Hello, Tell me your name again!</title>");
  out.println("</head>");
  out.println("<body>");
  out.println("<h2>Hello, " + name + "</h2> <br>");

  out.print("<form action=\"");
  out.println("NamedHello\" method=POST>");
  out.println("<input type=text length=20 name=yourname><br>");
  out.println("<input type=submit></form>");

  out.println("</body>");
  out.println("</html>");
  }
}
```

Servlet/NetBean

- Có hướng dẫn cài apache
  - http://www.cs.wcupa.edu/~rkline/Java/servlets-jsp-netbeans.html
- Có hình ảnh minh họa
  - http://www.java-tips.org/java-tutorials/tutorials/introduction-to-java-servlets-with-netbeans.html

Bài tập 2

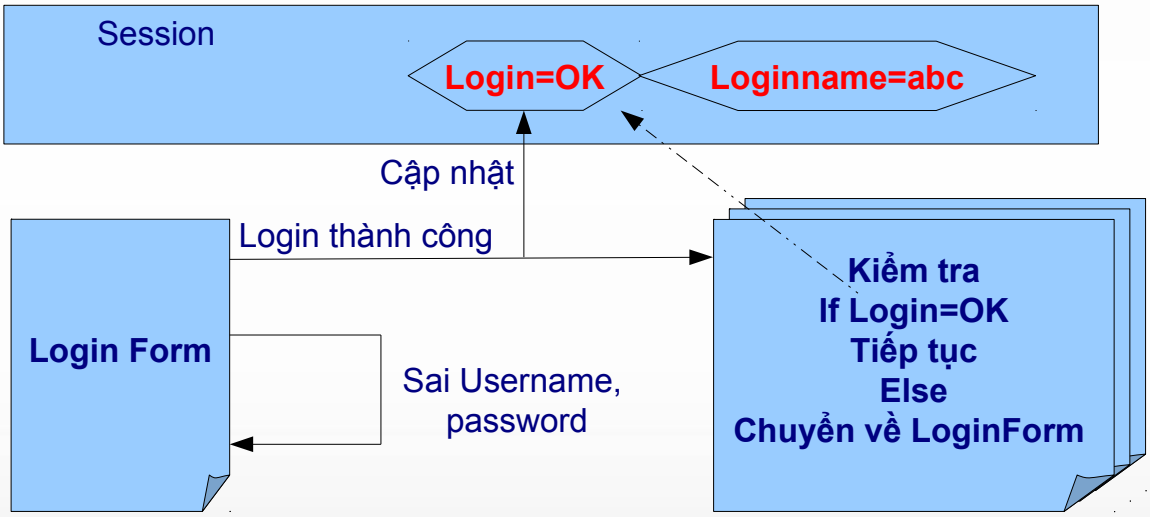
- Hãy viết chương trình tạo form nhập vào các thông tin sau:
  - Họ tên sinh viên
  - Mã số sinh viên
  - Năm sinh
- Nếu thông tin nào thiếu thì yêu cầu người dùng nhập lại, các thông tin đã nhập rồi thì hiển thị ngược lại trong form để người dùng không phải nhập lại
- In lại các thông tin nhận được đầy đủ

Bài tập 3

- Hãy viết chương trình tạo form nhập vào các thông tin sau:
  - Họ tên sinh viên
  - Mã số sinh viên
  - Năm sinh
- Nếu thông tin nào thiếu thì yêu cầu người dùng nhập lại, các thông tin đã nhập rồi thì hiển thị ngược lại trong form để người dùng không phải nhập lại
- Nhập các thông tin này vào cơ sở dữ liệu

Session

- Phương thức chia sẻ dữ liệu giữa các trang web trong một phiên làm việc



HttpSession

- Là giao diện được cung cấp bởi Servlet cho phép nhận dạng một người qua nhiều yêu cầu truy cập và lưu trữ thông tin về người này
- Được tạo ra từ phương thức getSession() của HttpServletRequest
  - HttpSession session = request.getSession();
  - Hàm này cần được gọi trước khi gởi về cho Client bất cứ dữ liệu gì

Phương thức của HttpSession

- public void setAttribute(String name, Object value): Gán đối tượng value vào session với tên là name
- public Object getAttribute(String name): Lấy lại đối tượng có tên là name đã được lưu giữ bởi session
- public Enumeration getAttributeNames()fl: Lấy danh sách tất cả các tên của tất cả các đối tượng được lưu trong session
- public String getId(): Trả về chuỗi định danh của session

Ví dụ về session

- Tạo dự án có tên là session-example
- Tạo một Servlet có tên là SessionTrack có phương thức doPost() như ví dụ tại địa chỉ http://www.tutorialspoint.com/servlets/servlets-session-tracking.htm
- Triển khai dự án này
- Truy cập đến dự án: http://localhost:8080/session-example/SessionTrack

Bài tập 4

- Tạo bảng Account lưu thông tin về tài khoản
  - Username, password
  - Nhập một số tài khoản vào bảng
- Viết servlet Login
  - Có form đăng nhập
  - Kiểm tra username/password dựa vào bảng Account
  - Nếu thành công in link đến Servlet StudentManager
  - Nếu thất bại, yêu cầu đăng nhập lại

Bài tập 4 (tt)

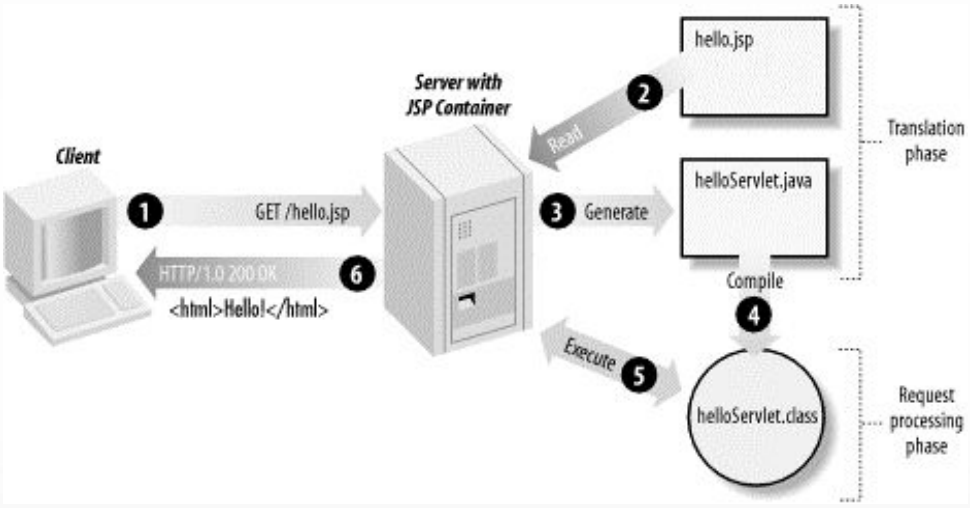
- Viết Servlet StudentManager
  - Kiểm tra xem người dùng đã login chưa,
  - Nếu đã login,
    - Hiển thị form nhập thông tin sinh viên
    - Cập nhật vào cơ sở dữ liệu
  - Nếu chưa đăng nhập
    - Gọi servlet Login để đăng nhập

JSP
Java Server Page

Servlet & JSP

- Một servlet tạo ra một trang web để đáp ứng cho một yêu cầu của một Client
- Servlet dùng Java để in ra mã HTML
  - Khó đọc, mất nhiều công sức nếu trang chứa nhiều văn bản
- JSP (Java Server Pages) là một cách để tạo ra các servlets
  - Viết như một trang HTML, cho phép trộn các mã Java để sinh nội dung động tùy theo yêu cầu của Client
  - Mã Java bao bọc trong thẻ đặc biệt <% …. %>
- Tập tin JSP có phần mở rộng là .jsp
  - JSP được dịch thành một servlet khi được biên dịch để chạy như các servlets đuowc viết thông thường
  - Browser/client chỉ nhận được mã HTML

Cơ chế của JSP



Servlet

```
public void doGet(HttpServletRequest request,
    HttpServletResponse response)    throws ServletException, IOException  {
    // Set response content type
    response.setContentType("text/html");
    // Actual logic goes here.
    PrintWriter out = response.getWriter();
    java.util.Date today = new java.util.Date()
    out.println("<HTML> <BODY>");
    out.println("Hello!  The time is now "+ today);
    out.println("</BODY> </HTML>");
 }
```

JSP

```
<HTML>
<BODervletY>
Hello!  The time is now <%= new java.util.Date() %>
</BODY>
</HTML>
```


Các thành phần của JSP

- <%= *expression* %>
  - Biểu thức *expression* được định trị và giá trị được hiển thị tại vị trí của thẻ
- <% *code* %>
  - *code* là các mã Java
  - Được gọi là các scriptlet

<%! *declarations* %>

- *declarations* là các khai báo biến hoặc định nghĩa phương thức

http://www.tutorialspoint.com/jsp/jsp_syntax.htm