TRƯỜNG ĐẠI HỌC
SƯ PHẠM KỸ THUẬT TP. HỒ CHÍ MINH
HCMC University of Technology and Education
HCMUTE

KHOA CÔNG NGHỆ THÔNG TIN
BỘ MÔN CÔNG NGHỆ PHẦN MỀM

LẬP TRÌNH WEB (WEBPR330479

Giới Thiệu JAVA SERVLET

THS. NGUYỄN HỮU TRUNG

PR33
WEB
0479

Thông tin Giảng viên

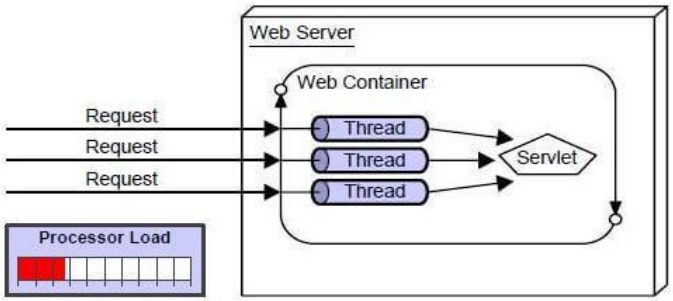
- Ths. Nguyễn Hữu Trung
- Khoa Công Nghệ Thông Tin
- Trường Đại học Sư Phạm Kỹ Thuật TP.HCM
- 090.861.7108
- trungnh@hcmute.edu.vn
- https://www.youtube.com/@baigiai

Nội dung bài học

- Java Servlet là gì?
- Kiến trúc của Servlet
- Nhiệm vụ của Servlet
- Hoạt động của Servlet
- Thiết lập môi trường Servlet
- Chu kỳ sống của Servlet
- Tạo ứng dụng Servlet trên Eclipse
- Xử lý Form trong Servlet
- Request và Response trong Servlet
- Cookie và Session trong Servlet
- PageRedirect trong Servlet

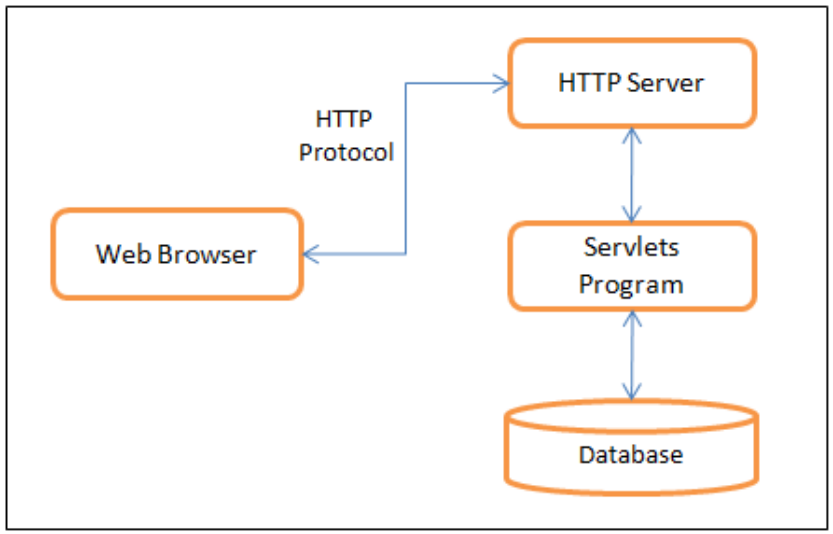
Java Servlet là gì?

- Servlet là bộ thư viện của Java được sử dụng để tạo ra ứng dụng web.
- Servlet API hỗ trợ nhiều Interface (Servlet, GenericServlet, HttpServlet, ServletRequest, ServletResponse) và Class cho lập trình java web.
- Java Servlets có các lợi thế sau:
  - Hiệu năng tốt vì mỗi thread cho mỗi request.
  - Servlets thực thi bên trong không gian địa chỉ của một Web server, không cần thiết phải tạo một tiến trình riêng biệt để xử lý mỗi request từ Client.
  - Servlets là độc lập trên nền tảng bởi vì chúng được viết bằng Java.
  - Servlets là đáng tin cậy và bảo mật.
  - Tính năng đầy đủ của thư viện của các lớp trong Java là luôn luôn có sẵn cho Servlets.
  - Bộ máy ảo JVM mạnh mẽ giúp quản lý bộ nhớ và xóa rác.



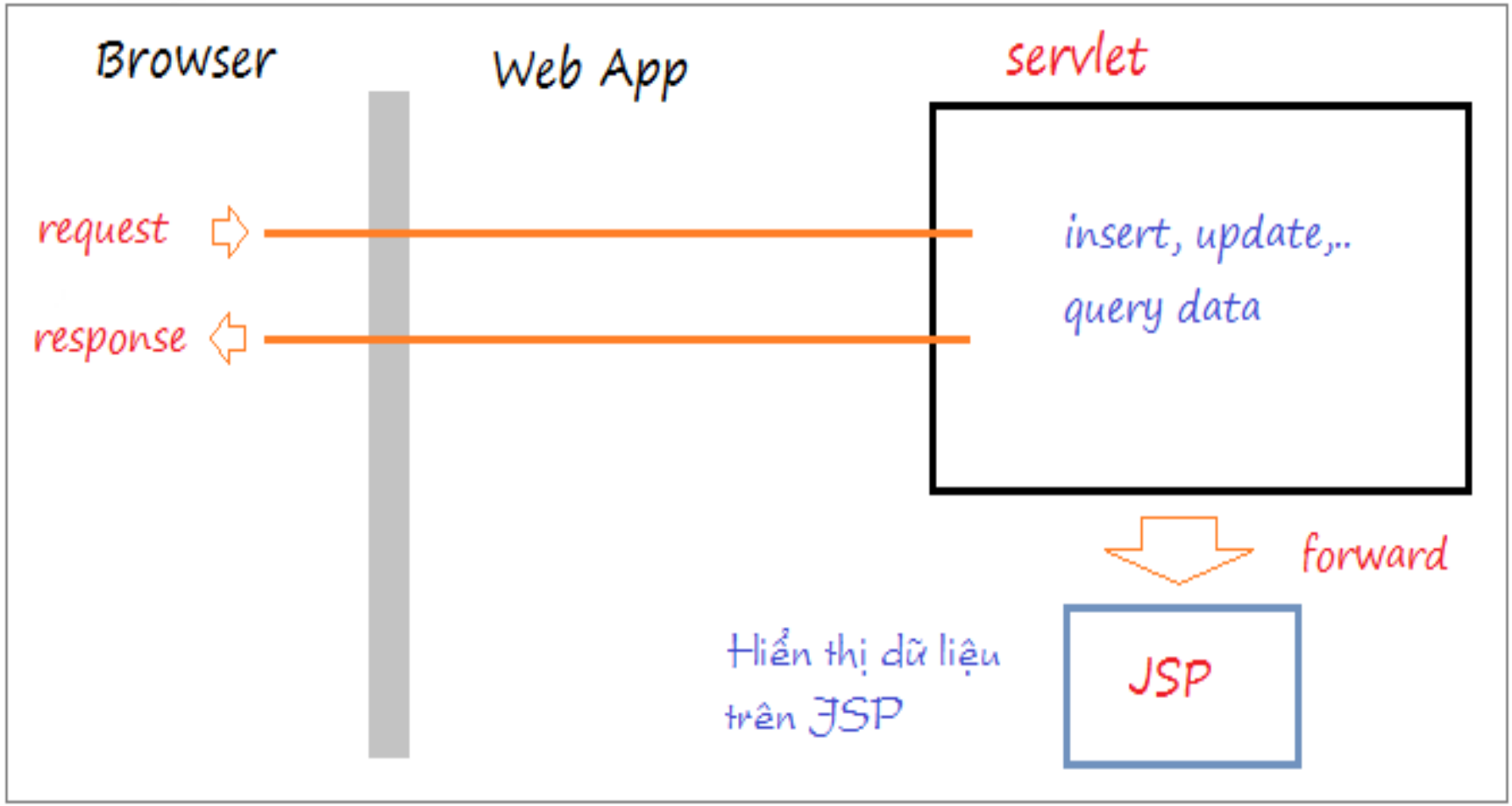
Nhiệm vụ Java Servlet trong ứng dụng Web

- Đọc dữ liệu do client gửi (HTML Form, Applet,..).
- Đọc dữ liệu yêu cầu HTTP ẩn được gửi bởi client (cookie, loại phương tiện truyền thông và các chương trình nén…)
- Xử lý dữ liệu và tạo ra các kết quả. Quá trình này có thể yêu cầu với một cơ sở dữ liệu.
- Gửi dữ liệu tới client (bao gồm văn bản (HTML hoặc XML), nhị phân (hình ảnh GIF), Excel, v.v…)
- Gửi phản hồi HTTP ẩn cho client (cookie, Caching)



Hoạt động của Java Servlet

- Java Servlet xử lý logic các tác vụ của web



Các gói Java Servlet

- Java Servlets là các lớp Java chạy bởi một web Server.
- Servlets có thể được tạo ra bằng cách sử dụng các **gói javax.servlet** và **javax.servlet.http** theo hướng **J2EE** hoặc **jakarta.servlet** theo hướng **Jakarta** Các lớp này thực hiện các đặc tả Java Servlet và JSP.
- Các Java Servlet đã được tạo ra và biên dịch giống như các lớp khác. Sau khi cài đặt gói servlet và thêm vào Classpath của máy tính, bạn có thể biên dịch các servlet bằng trình dịch Java của JDK hoặc bất kỳ trình biên dịch nào khác.

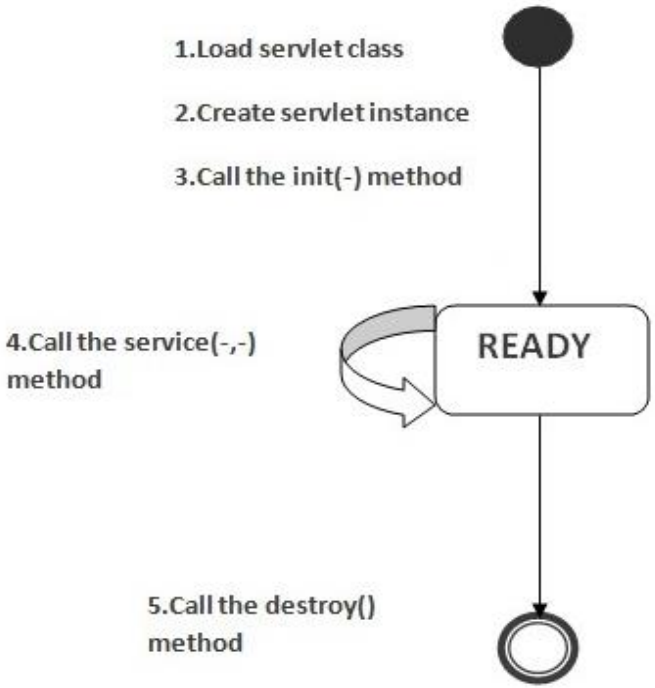
Thiết lập môi trường Servlet

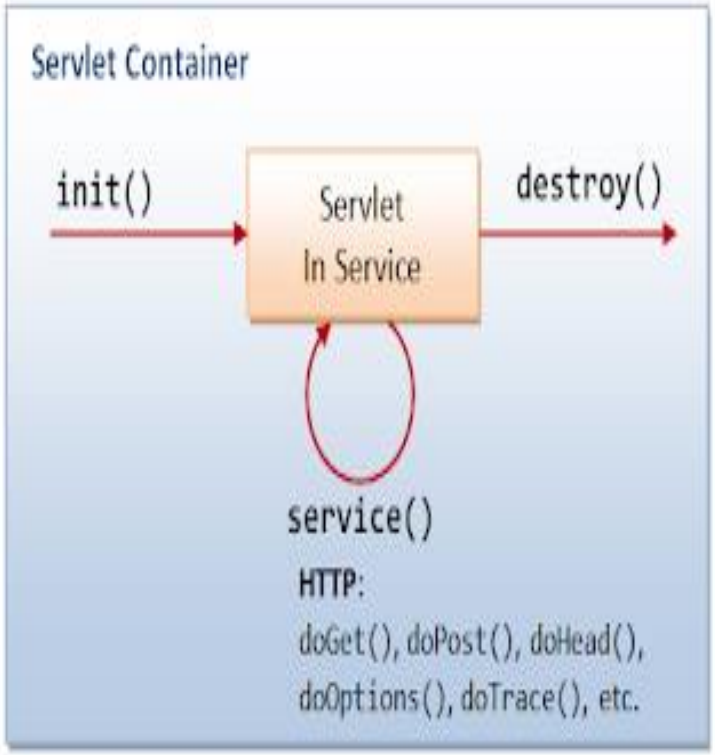
- **Cài đặt môi trường Servlet** bao gồm các bước sau:
  - Thiết lập bộ phát triển Java (Java Development Kit-JDK):
  - Thiết lập Web Server – Tomcat
  - Thiết lập quản lý project theo Maven

  => Xem lại bài trước về cách cài đặt và cấu hình môi trường tại đây.

Chu kỳ sống của Java Servlet

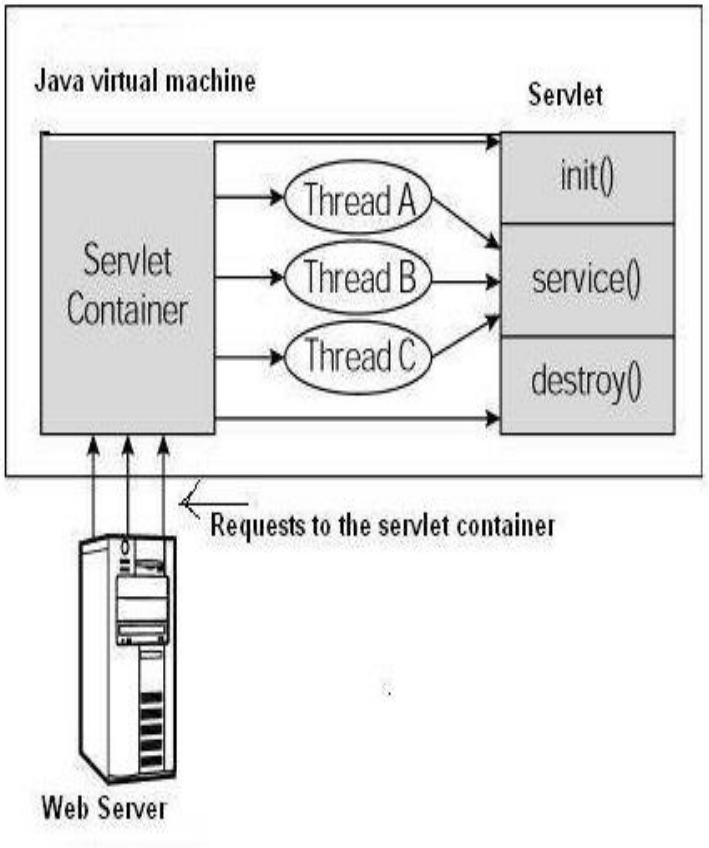
- Một **chu kỳ của Java Servlet** là toàn bộ quá trình từ khi tạo ra đến khi bị hủy. Sau đây là tổng quan về vòng đời của servlet:
  - Servlet được khởi tạo bằng cách gọi phương thức **init ()**.
  - Phương thức servlet **service()** được gọi để xử lý yêu cầu của khách hàng.
  - Servlet được hủy bằng cách gọi phương thức **destroy()**.
  - Cuối cùng, servlet được thu thập bởi bộ sưu tập Garbage Collector của JVM.





Sơ đồ kiến trúc

- Hình dưới đây mô tả vòng đời của servlet điển hình:
  - Đầu tiên các HTTP Request đến máy chủ được ủy quyền cho Container của Servlet.
  - Container Servlet tải servlets trước khi gọi phương thức service().
  - Sau đó, Container Servlet xử lý nhiều yêu cầu bằng cách sinh ra nhiều Thread(luồng), mỗi luồng thực hiện service() cho một đối tượng servlet đơn.

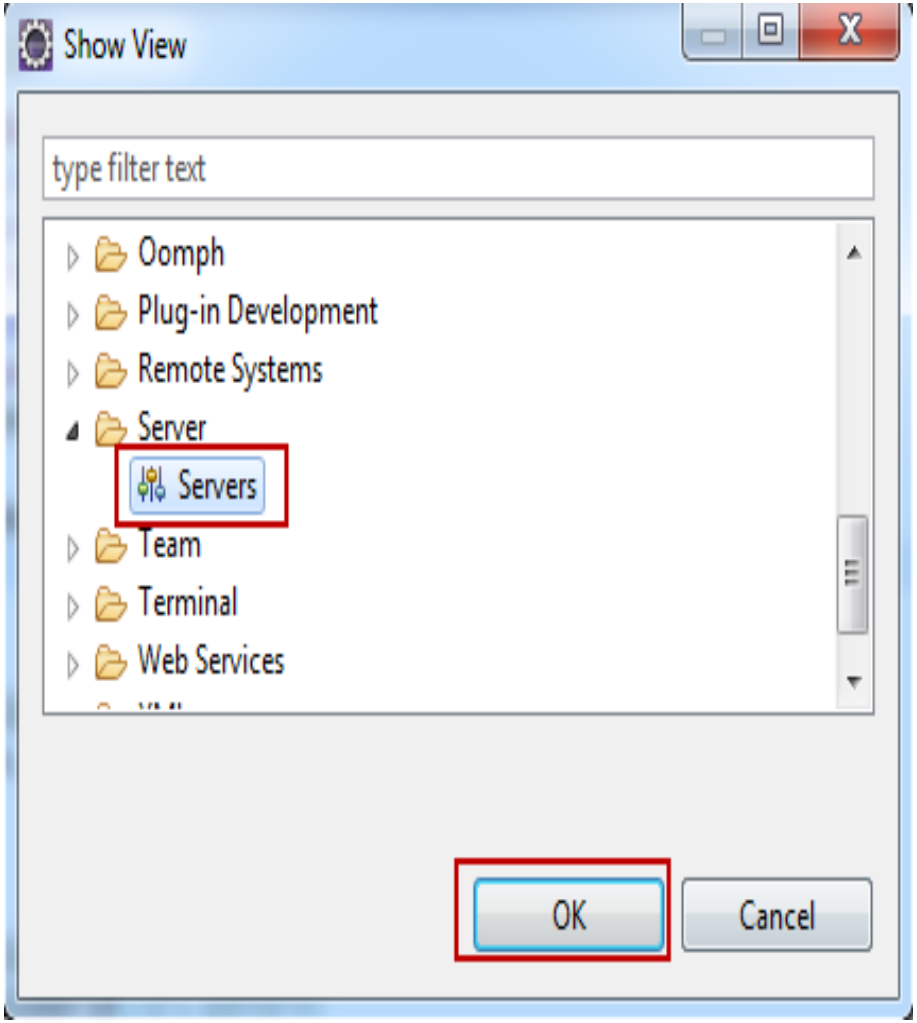


Tạo ứng dụng Servlet trên Eclipse

- Các bước tạo ứng dụng Servlet trên eclipse:
  - Cấu hình Tomcat trên eclipse.
  - Tạo project “Maven Project" trên eclipse.
  - Tạo lớp Servlet.
  - Cấu hình Servlet.
  - Chạy ứng dụng.

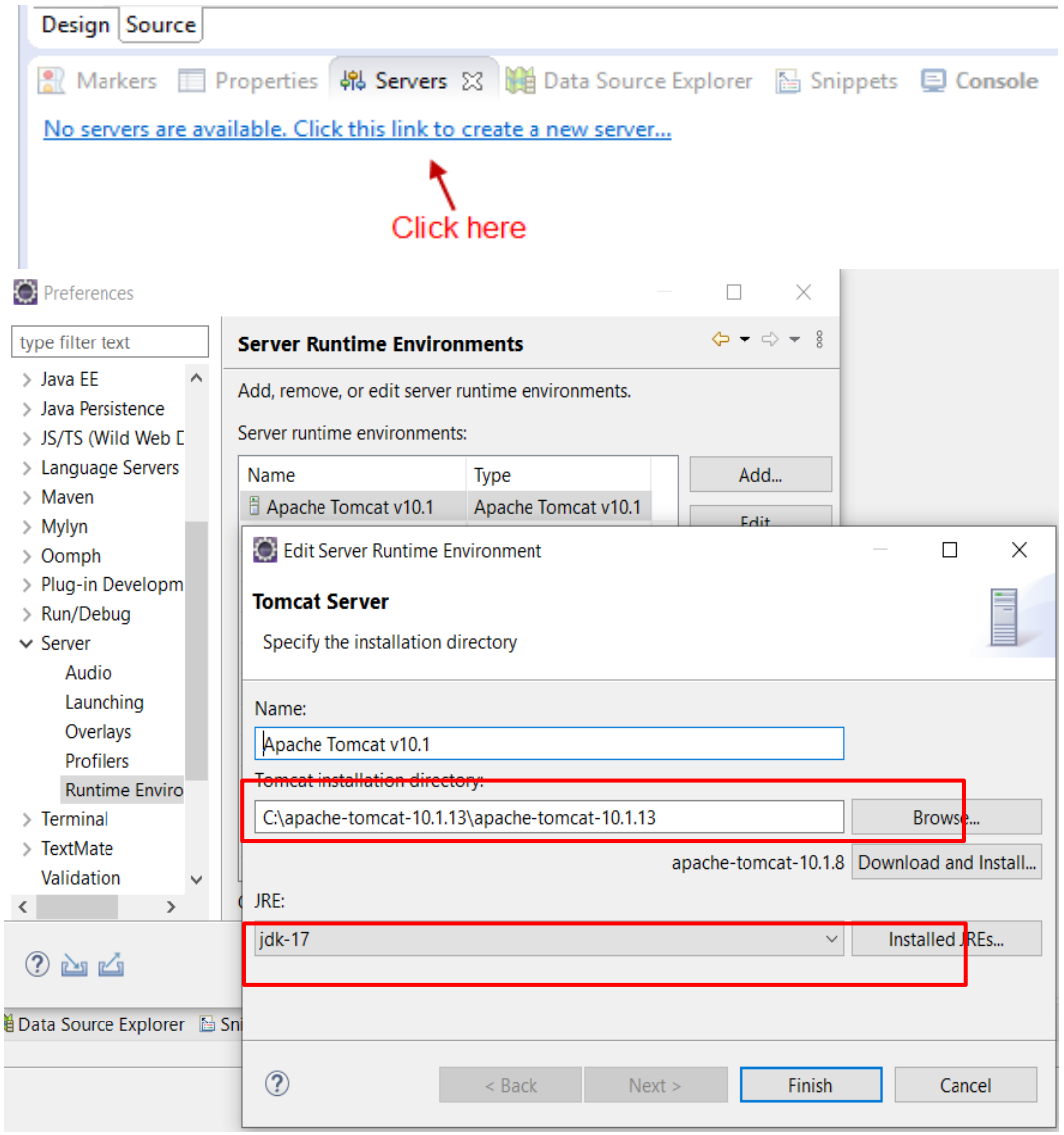
Tạo ứng dụng Servlet trên Eclipse

- Gọi Tomcat trên eclipse
  - Chọn Window -> Show View -> Other -> Server -> Click OK.



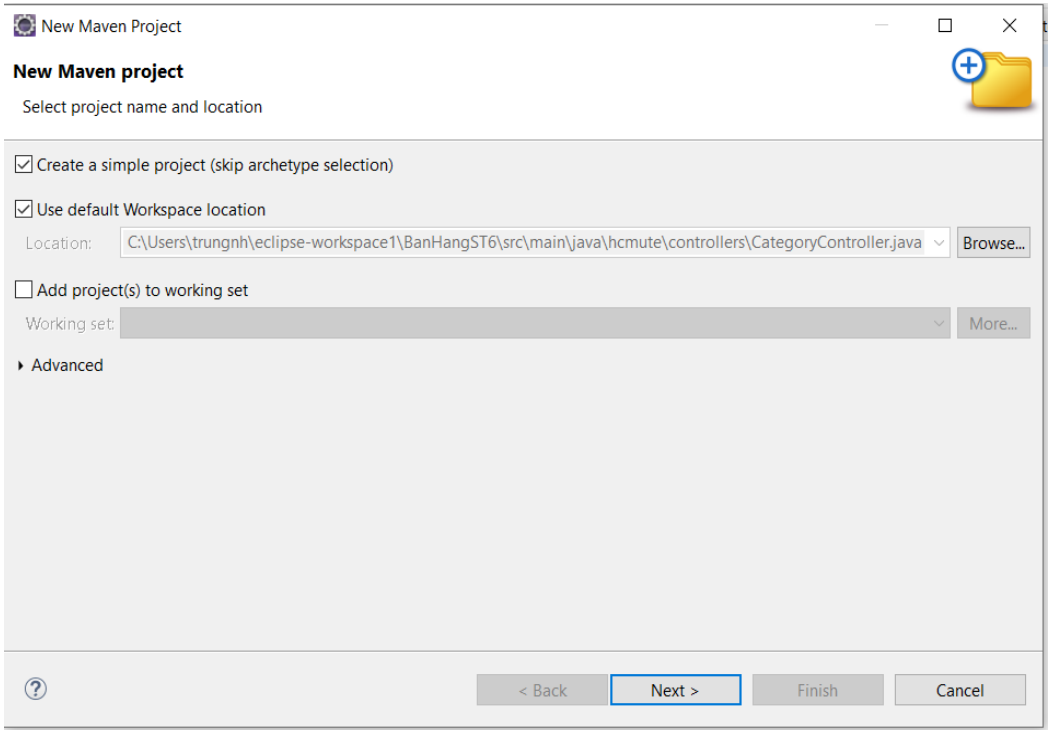
Tạo ứng dụng Servlet trên Eclipse

- Gọi Tomcat trên eclipse
  - Click vào link như trong hình sau:
  - Chọn Tomcat Server -> click Add.
  - Nhập Tomcat installation direction là đường dẫn đến thư mục tomcat server trên máy tính của bạn:



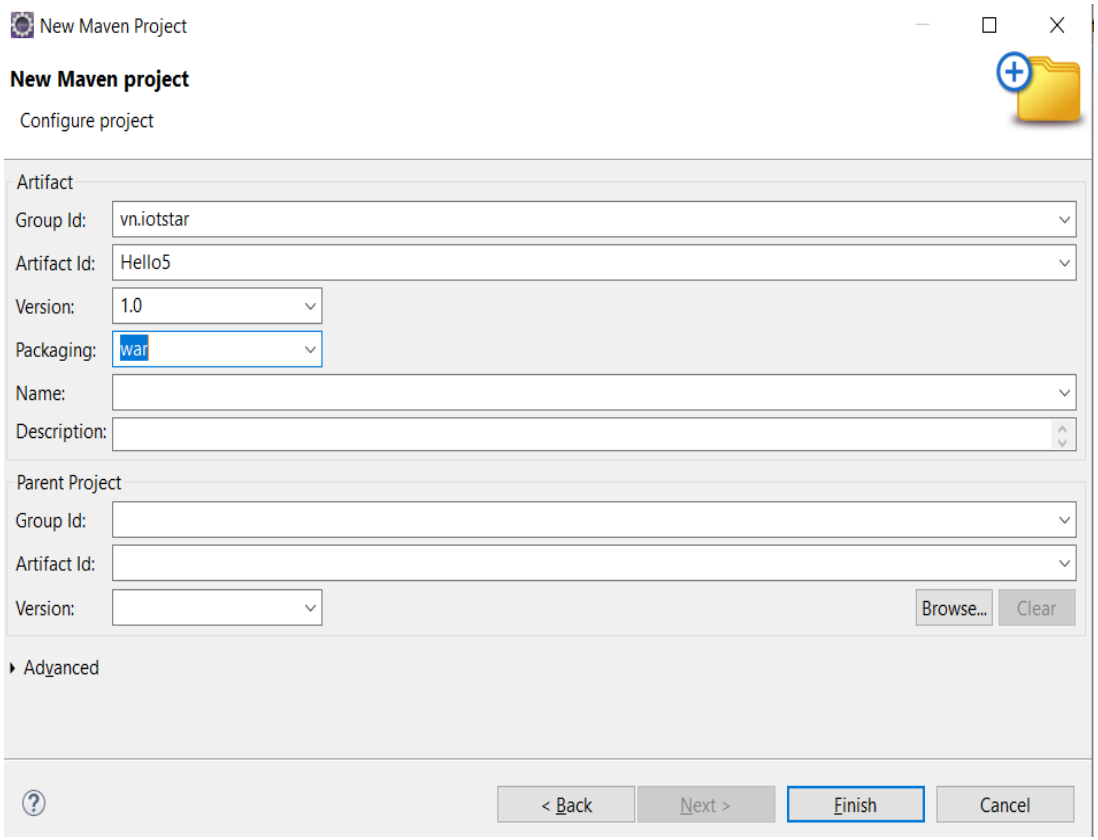
Tạo ứng dụng Servlet trên Eclipse

- Tạo project “Maven Project" trên eclipse
  - Chọn File -> New -> Maven Project
  - Tích vào checkbox Create a simple project.
  - Click Next.



Tạo ứng dụng Servlet trên Eclipse

- Tạo project “Maven Project" trên eclipse
  - Đặt tên Group ID
  - Đặt tên project trong Artifact ID
  - Gõ Version
  - Chọn kiểu đóng gói trong packing như hình
  - Bấm Finish
  - Trước khi bấm finish phải kiểm tra kết nối Internet.



Tạo ứng dụng Servlet trên Eclipse

- Cấu hình file pom.xml (file quản lý thư viện trong maven project)
  - Cấu hình build và cấu hình dependencies(nơi chứa các thư viện) trong file pom.xml

```
https://maven.apache.org/xsd/maven-4.0.0.xsd (xsi:schemaLocation)
<project xmlns="http://maven.apache.org/POM/4.0.0"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 https://maven.apache.org/xsd/maven-4.0.0.xsd">
    <modelVersion>4.0.0</modelVersion>
    <groupId>iotstar.vn</groupId>
    <artifactId>Hello</artifactId>
    <version>1.0</version>
    <packaging>war</packaging>
    <dependencies>
        <!-- nơi chứa thư viện
    </dependencies>
    <build>
        <pluginManagement>
            <plugins>
                <plugin>
                    <groupId>org.apache.maven.plugins</groupId>
                    <artifactId>maven-compiler-plugin</artifactId>
                    <version>3.11.0</version>
                    <configuration>
                        <release>17</release>
                    </configuration>
                </plugin>
                <plugin>
                    <groupId>org.apache.maven.plugins</groupId>
                    <artifactId>maven-war-plugin</artifactId>
                    <version>3.4.0</version>
                </plugin>
            </plugins>
        </pluginManagement>
    </build>
</project>
```

Bỏ vào

```
<dependency>
<groupId>javax.servlet</groupId>
<artifactId>javax.servlet-
api</artifactId>
<version>4.0.1</version>
<scope>provided</scope>
</dependency>
<dependency>

<groupId>jakarta.servlet</groupId>
<artifactId>jakarta.servlet-
api</artifactId>
    <version>6.1.0</version>
    <scope>provided</scope>
</dependency>
```

Tạo ứng dụng Servlet trên Eclipse

- Tạo lớp Servlet
  - Tạo lớp Servlet bằng cách extends lớp **javax.servlet.http.HttpServlet** của **J2EE** hoặc **jakarta.servlet.http.HttpServlet** của **Jakarta**, một lớp trừu tượng thực hiện giao diện Servlet và được thiết kế đặc biệt để xử lý các yêu cầu HTTP. Ở tất cả hướng dẫn sẽ sử dụng **jakarta.servlet.http.HttpServlet** của **Jakarta**

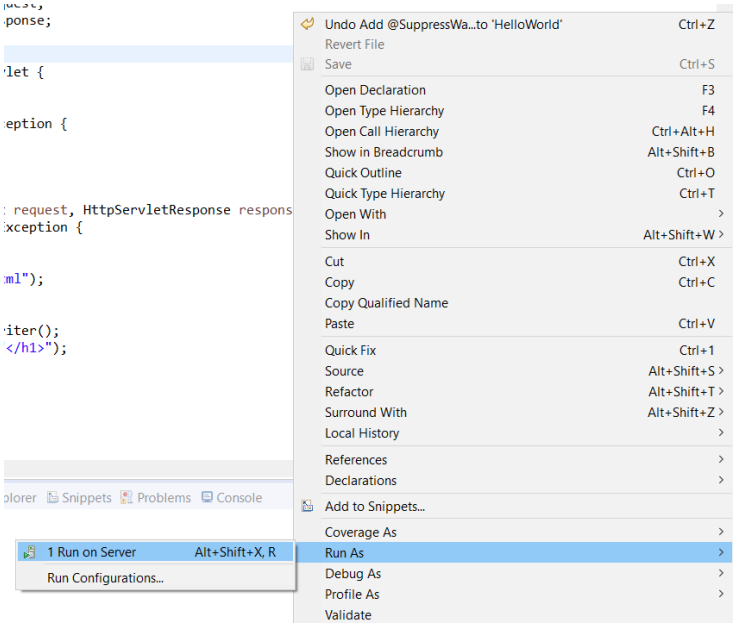
```
public class HelloServlet extends HttpServlet {
    private String message;
   @SuppressWarnings("serial")
    public void init() throws ServletException {
        // Do required initialization
        message = "Hello World";
    }
    public void doGet(HttpServletRequest request, HttpServletResponse response)
            throws ServletException, IOException {

        // Set response content type
        response.setContentType("text/html");

        // Actual logic goes here.
        PrintWriter out = response.getWriter();
        out.println("<h1>" + message + "</h1>");
    }
    public void destroy() {
        // do nothing.
    }
}
```

Tạo ứng dụng Servlet trên Eclipse

- Cấu hình URL cho Servlet (cấu hình bằng xml hoặc Annotation cho Servlet 3+ trở lên)
  - Cấu hình url bằng Annotation để truy cập servlet: sử dụng Annotation @WebServlet().

```
@WebServlet(urlPatterns = {"/hi"})
public class HelloServlet extends HttpServlet {
```

- Chạy ứng dụng
  - Để chạy servlet bạn làm như sau: click chuột phải vào servlet -> Run As -> Run On Server.. Rồi chọn Tomcat server 10.1+ nhé.



TRƯỜNG ĐẠI HỌC
SƯ PHẠM KỸ THUẬT TP. HỒ CHÍ MINH
HCMC University of Technology and Education
HCMUTE

KHOA CÔNG NGHỆ THÔNG TIN
BỘ MÔN CÔNG NGHỆ PHẦN MỀM

LẬP TRÌNH WEB (WEBPR330479

CẤU HÌNH URL JAVA SERVLET

THS. NGUYỄN HỮU TRUNG

PR33 WEB 0479

Thông tin Giảng viên

- Ths. Nguyễn Hữu Trung
- Khoa Công Nghệ Thông Tin
- Trường Đại học Sư Phạm Kỹ Thuật TP.HCM
- 090.861.7108
- trungnh@hcmute.edu.vn
- https://www.youtube.com/@baigiai

Cấu hình URL Servlet bằng XML

- Mở file web.xml để map class HelloServlet thành URL /hello như sau:

```
<servlet>
  <servlet-name>HelloServlet</servlet-name>
  <servlet-class>vn.iotstar.controller.HelloServlet</servlet-class>
</servlet>
<servlet-mapping>
  <servlet-name>HelloServlet</servlet-name>
  <url-pattern>/hello</url-pattern>
</servlet-mapping>
```

```
public class HelloServlet extends HttpServlet {
@Override
protected void doGet(HttpServletRequest req, HttpServletResponse
resp) throws ServletException, IOException {
resp.setContentType("text/html");
PrintWriter printW = resp.getWriter();
printW.println("<h1>HelloWrod</h1>");
printW.close();
```

Cấu hình URL Servlet bằng Annotation

- Sử dụng @WebServlet() trong class (chỉ dùng cho Servlet version >=3.0)

```
@WebServlet(urlPatterns= {"/hello","/xin-chao"})
public class HelloServlet extends HttpServlet {

@Override
protected void doGet(HttpServletRequest req, HttpServletResponse
resp) throws ServletException, IOException {

resp.setContentType("text/html");

PrintWriter printW = resp.getWriter();
printW.println("<h1>HelloWrod</h1>");

printW.close();
}
}
```

TRƯỜNG ĐẠI HỌC
SƯ PHẠM KỸ THUẬT TP. HỒ CHÍ MINH
HCMC University of Technology and Education
HCMUTE

KHOA CÔNG NGHỆ THÔNG TIN
BỘ MÔN CÔNG NGHỆ PHẦN MỀM

LẬP TRÌNH WEB (WEBPR330479

Đọc dữ liệu từ URL và FORM trong JAVA SERVLET

THS. NGUYỄN HỮU TRUNG

PR33 WEB 0479

GET URL trong Java Servlet

- Url get trên trình duyệt:
  http://localhost:8080/HelloServlet/hello?ten=NguyenHuuTrung
- Với tham số: ten được truyền vào Servlet
- Trong Servlet nhận tham số tên bằng
  - //Nhận tham số từ request
  - String ten = request.getParameter("ten");
- Hiển thị kết quả cho trang web
  - //hiển thị lên trang bằng đối tượng PrintWriter()
  - printWriter.println("Xin chao " + ten);

Xử lý Form trong Servlet

- **Phương thức GET**
  - Phương thức GET gửi thông tin người dùng được mã hoá được nối vào yêu cầu trang. Trang và thông tin được mã hoá được tách biệt bằng **?** (dấu chấm hỏi) như sau:

    http://www.test.com/hello?key1=value1&key2=value2
  - Không bao giờ sử dụng phương thức GET nếu bạn có mật khẩu hoặc thông tin nhạy cảm khác để chuyển đến máy chủ. Phương thức GET có giới hạn kích thước: chỉ có 1024 ký tự có thể được sử dụng trong một request.
  - Thông tin này được truyền bằng cách sử dụng tiêu đề QUERY_STRING và sẽ có thể truy cập qua biến môi trường QUERY_STRING và Servlet sẽ xử lý loại yêu cầu này bằng cách sử dụng phương thức **doGet()**.
- **Phương thức POST**
  - Phương thức này gói thông tin theo cách chính xác giống như phương thức GET, nhưng thay vì gửi nó như một chuỗi văn bản sau một ? (dấu chấm hỏi) trong URL thì phương thức này gửi nó như một thông điệp riêng biệt. Thông báo này đi kèm với chương trình backend dưới dạng đầu vào tiêu chuẩn mà bạn có thể phân tích và sử dụng cho quá trình xử lý của bạn.
  - Servlet xử lý kiểu yêu cầu này sử dụng **phương thức doPost()**.

Xử lý Form trong Servlet

- Servlet xử lý dữ liệu từ một Form bằng cách sử dụng các phương thức sau đây tùy thuộc vào tình huống:
  - **getParameter()** - Gọi phương thức **request.getParameter()** để lấy giá trị của một tham số của form.
  - **getParameterValues()** - Gọi phương thức này nếu tham số xuất hiện nhiều lần và trả về nhiều giá trị, ví dụ checkbox.
  - **getParameterNames()** - Gọi phương thức này nếu bạn muốn có một danh sách đầy đủ của tất cả các tham số trong yêu cầu hiện tại.

Ví dụ phương thức GET sử dụng URL

- Tạo servlet **HelloForm** để xử lý yêu cầu từ máy khách (HelloForm.java)

```
@WebServlet(urlPatterns= {"/helloform"})
public class HelloForm extends HttpServlet {

public void doGet(HttpServletRequest request, HttpServletResponse response)
            throws ServletException, IOException {

        //Nhận dữ liệu từ request URL
         String ten = request.getParameter("ten");
         String holot = request.getParameter("holot");

         //Hiển thị dữ liệu ra web bằng đối tượng PrintWriter
         PrintWriter out = response.getWriter();
         out.println("<b>First Name</b>: " + ten + "<br/><b>Last Name</b>:
"+ holot);

}
```

http://localhost:8080/HelloServlet/helloform?holot=NguyenHuu&ten=Trung

Ví dụ phương thức GET sử dụng Form

- Tạo trang **Index.html** trong thư mục Webapp

```
<form action="helloform" method="GET">
    Tên: <input type="text" name="ten"> <br />
    Họ lót: <input type="text" name="holot" />
    <input type="submit" value="Submit" />
  </form>
```

- HelloForm.java

```
@Override
protected void doPost(HttpServletRequest request, HttpServletResponse
response) throws ServletException, IOException {
 // Set response content type
        response.setContentType("text/html");
        response.setCharacterEncoding("UTF-8");
        request.setCharacterEncoding("UTF-8");
       //Nhận dữ liệu từ request URL
        String ten = request.getParameter("ten");
        String holot = request.getParameter("holot");
        //Hiển thị dữ liệu ra web bằng đối tượng PrintWriter
        PrintWriter out = response.getWriter();
out.println("<b>First Name</b>: " + ten + "<br/><b>Last Name</b>: "+ holot);
}
```

Ví dụ phương thức POST sử dụng Form

- Tạo trang **Index.html** trong thư mục Webapp

```
<form action="helloform" method="POST">
    Tên: <input type="text" name="ten"> <br />
    Họ lót: <input type="text" name="holot" />
    <input type="submit" value="Submit" />
  </form>
```

- HelloForm.java

```
@Override
protected void doPost(HttpServletRequest request, HttpServletResponse
response) throws ServletException, IOException {
 // Set response content type
        response.setContentType("text/html");
        response.setCharacterEncoding("UTF-8");
        request.setCharacterEncoding("UTF-8");
       //Nhận dữ liệu từ request URL
        String ten = request.getParameter("ten");
        String holot = request.getParameter("holot");
        //Hiển thị dữ liệu ra web bằng đối tượng PrintWriter
        PrintWriter out = response.getWriter();
out.println("<b>First Name</b>: " + ten + "<br/><b>Last Name</b>: "+ holot);
}
```

Ví dụ truyền dữ liệu CheckBox trong Servlet

- **CheckBoxAction.java**

```
public void doGet(HttpServletRequest request,
HttpServletResponse response)
        throws ServletException, IOException {
        // Set response content type
        response.setContentType("text/html");
        PrintWriter out = response.getWriter();
        String title = "Doc du lieu tu CheckBox trong Servlet";
        String docType =
           "<!doctype html public \"-//w3c//dtd html 4.0 " +
           "transitional//en\">\n";
```

Ví dụ truyền dữ liệu CheckBox trong Servlet

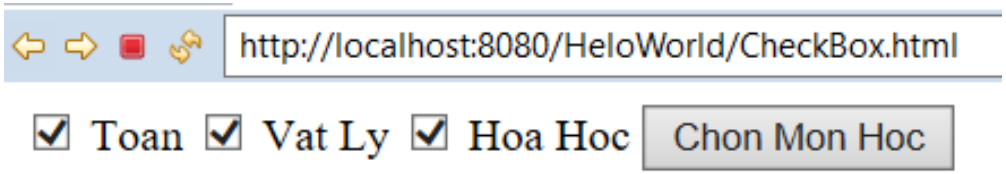
- **CheckBoxAction.java**

```
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>Toan : </b>: "
                   + request.getParameter("toan") + "\n" +
                   "  <li><b>Vat Ly: </b>: "
                   + request.getParameter("ly") + "\n" +
                   "  <li><b>Hoa Hoc: </b>: "
                   + request.getParameter("hoa") + "\n" +
                "</ul>\n" +
             "</body>" +
          "</html>"
       );
   }
```

Ví dụ truyền dữ liệu CheckBox trong Servlet

CheckBox.html trong webapp

```
<form action="CheckBoxAction" method="POST">
   <input type="checkbox" name="toan" checked="checked" /> Toan
   <input type="checkbox" name="ly" /> Vat Ly
   <input type="checkbox" name="hoa" /> Hoa Hoc
   <input type="submit" value="Chon Mon Hoc" />
 </form>
```



Ví dụ đọc tất cả Parameters của Form trong Servlet

- Sử dụng phương thức **getParameterNames()** của HttpServletRequest để đọc tất cả các tham số của HTML Form. Phương thức này trả về một Enumeration chứa các tên tham số theo một thứ tự không xác định.
- Đối với Enumeration, chúng ta có thể lặp Enumeration theo cách chuẩn bằng cách sử dụng phương thức hasMoreElements() để xác định khi nào dừng lại và sử dụng phương thức nextElement() để lấy từng tên tham số.

- **ReadParamsAction.java**

```
// Method to handle GET method request.
   public void doGet(HttpServletRequest request, HttpServletResponse response)
         throws ServletException, IOException {
      // Set response content type
      response.setContentType("text/html");
      PrintWriter out = response.getWriter();
      String title = "Reading All Form Parameters";
      String docType = "<!doctype html>\n";
```

Ví dụ đọc tất cả Parameters của Form trong Servlet

- **ReadParamsAction.java**

```
 out.println(docType +
             "<html>\n" +
             "<head><title>" + title + "</title></head>\n" +
             "<body bgcolor = \"#f0f0f0\">\n" +
             "<h1 align = \"center\">" + title + "</h1>\n" +
             "<table width = \"100%\" border = \"1\" align =
\"center\">\n" +
             "<tr bgcolor = \"#949494\">\n" +
                 "<th>Param Name</th>" +
                 "<th>Param Value(s)</th>\n"+
             "</tr>\n"
         );
         // get all parameters of form
         @SuppressWarnings("rawtypes")
Enumeration paramNames = request.getParameterNames();
```

Ví dụ đọc tất cả Parameters của Form trong Servlet

- **ReadParamsAction.java trong vn.iotstar**

```
// read parameters
        while (paramNames.hasMoreElements()) {
            String paramName = (String) paramNames.nextElement();
            out.print("<tr><td>" + paramName + "</td>\n<td>");
            String[] paramValues =
request.getParameterValues(paramName);

            // Read single valued data
            if (paramValues.length == 1) {
                String paramValue = paramValues[0];
                if (paramValue.length() == 0)
                    out.println("<i>No Value</i>");
                else
                    out.println(paramValue);
            } else {
```

Ví dụ đọc tất cả Parameters của Form trong Servlet

- **ReadParamsAction.java trong vn.iotstar**

```
// Read multiple valued data
        out.println("<ul>");
        for (int i = 0; i < paramValues.length; i++) {
                    out.println("<li>" + paramValues[i]);
                }
                out.println("</ul>");
            }
        }
        out.println("</tr>\n</table>\n</body></html>");
    }
```

Ví dụ đọc tất cả Parameters của Form trong Servlet

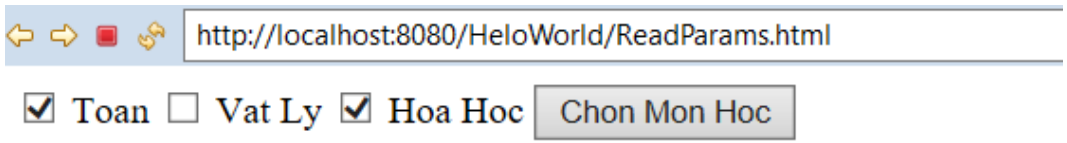
- **ReadParamsAction.java trong vn.iotstar**

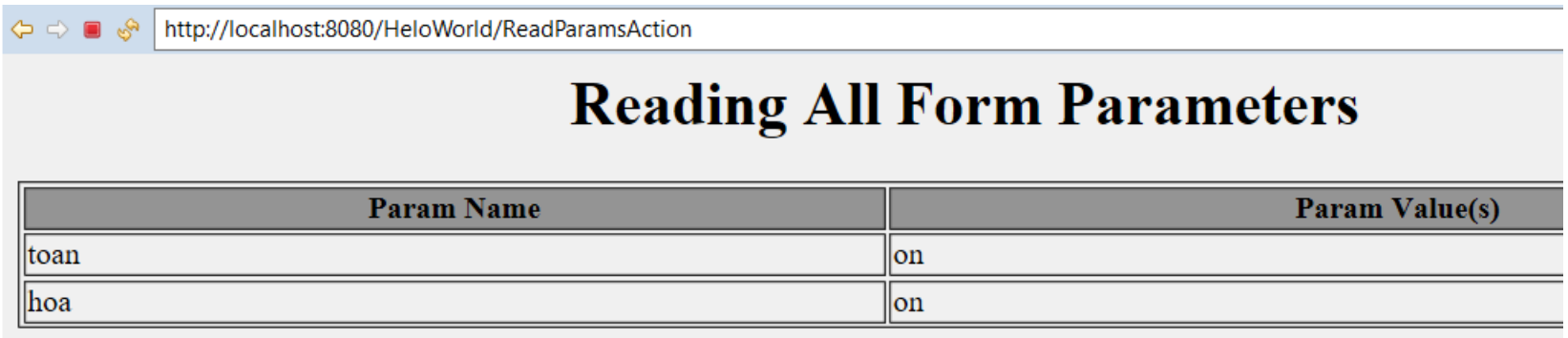
```
// Read multiple valued data
                out.println("<ul>");

                for (int i = 0; i < paramValues.length; i++) {
                    out.println("<li>" + paramValues[i]);
                }
                out.println("</ul>");
            }
        }
        out.println("</tr>\n</table>\n</body></html>");
    }
```

Ví dụ truyền dữ liệu CheckBox trong Servlet

ReadParams.html trong webapp

```
<form action="ReadParamsAction" method="POST">
   <input type="checkbox" name="toan" checked="checked" /> Toan
   <input type="checkbox" name="ly" /> Vat Ly
   <input type="checkbox" name="hoa" /> Hoa Hoc
   <input type="submit" value="Chon Mon Hoc" />
 </form>
```





TRƯỜNG ĐẠI HỌC
SƯ PHẠM KỸ THUẬT TP. HỒ CHÍ MINH
HCMC University of Technology and Education
HCMUTE

KHOA CÔNG NGHỆ THÔNG TIN
BỘ MÔN CÔNG NGHỆ PHẦN MỀM

LẬP TRÌNH WEB (WEBPR330479

ServletConfig, ServletConfig trong Java Servlet

THS. NGUYỄN HỮU TRUNG

PR33
WEB
0479

Thông tin Giảng viên

- Ths. Nguyễn Hữu Trung
- Khoa Công Nghệ Thông Tin
- Trường Đại học Sư Phạm Kỹ Thuật TP.HCM
- 090.861.7108
- trungnh@hcmute.edu.vn
- https://www.youtube.com/@baigiai

ServletConfig trong Java Servlet

- Khi **Web Container** khởi tạo một Servlet, nó tạo một đối tượng **ServletConfig** cho Servlet. Đối tượng ServletConfig được sử dụng để truyền thông tin tới một Servlet trong quá trình khởi tạo bằng cách lấy thông tin cấu hình từ **web.xml**.
- **Các phương thức ServletConfig**
  - String getInitParameter(String name): trả về một tham số khởi tạo giá trị String, hoặc NULL nếu tham số không tồn tại.
  - Enumeration getInitParameterNames(): trả về tên của các tham số khởi tạo của servlet như là một Enumeration of String objects, hoặc một Enumeration trống nếu servlet không có tham số khởi tạo.
  - ServletContext getServletContext(): trả về một tham chiếu đến ServletContext
  - String getServletName(): trả về tên của ví dụ servlet

ServletConfig trong Java

- Khởi tạo và thực thi một ServletConfig:
  - B1: cấu hình trong web.xml hoặc trong Servlet

```
<servlet>
      <servlet-name>MyServlet</servlet-
name>
      <servlet-
class>vn.iotstar.MyServlet</servlet-
class>
     <init-param>
         <param-name>email</param-name>
         <param-
value>trungnh@hcmute.edu.vn</param-
value>
      </init-param>
    </servlet>
```

```
@WebServlet(urlPatterns=
{"/check-config"},
initParams={@WebInitParam(name
="name",value="Nguyễn Hữu
Trung"),@WebInitParam(name="em
ail",value="trungnh")})
```

B2: Gọi bên trong lớp Servlet:

```
ServletConfig sc = getServletConfig();
out.println(sc.getInitParameter("email"));
```

ServletContext trong Java Servlet

- ServletContext() là đối tượng chứa thông tin chung của servlet cho toàn bộ web.
- getServletContext(key,value)
  - //ServletContext
  - getServletContext().setAttribute("name", "Nguyễn Hữu Trung");
  - //Lấy dữ liệu ServletContext()
  - String name = (String) getServletContext().getAttribute("name");

ServletContext trong Servlet Java

- 2. Cấu hình định nghĩa ServletContext() trong web.xml
  - <context-param>
    - <param-name>jdbc</param-name>
    - <param-value>mysql</param-value>
  - </context-param>
- Gọi ServletContext() trong Servlet
  - //Gọi ServletContext() trong web.xml
  - String jdbc = getServletContext().getInitParameter("jdbc");

TRƯỜNG ĐẠI HỌC
SƯ PHẠM KỸ THUẬT TP. HỒ CHÍ MINH
HCMC University of Technology and Education
HCMUTE

KHOA CÔNG NGHỆ THÔNG TIN
BỘ MÔN CÔNG NGHỆ PHẦN MỀM

LẬP TRÌNH WEB (WEBPR330479

ServletRequest, ServletResponse trong Java Servlet

THS. NGUYỄN HỮU TRUNG

PR33
WEB
0479

Thông tin Giảng viên

- Ths. Nguyễn Hữu Trung
- Khoa Công Nghệ Thông Tin
- Trường Đại học Sư Phạm Kỹ Thuật TP.HCM
- 090.861.7108
- trungnh@hcmute.edu.vn
- https://www.youtube.com/@baigiai

ServletRequest trong Java Servlet

- Khi trình duyệt gửi yêu cầu (**request**) đến một trang web. Nó gửi rất nhiều thông tin đến web server nhưng những thông tin này không thể đọc trực tiếp vì chúng là một phần của Header của HTTP request.
- Các phương thức sau đây có thể được sử dụng để đọc HTTP Header trong chương trình servlet của bạn. Các phương thức này có sẵn trong đối tượng **HttpServletRequest**.

ServletRequest trong Java Servlet

TT	Phương pháp & Mô tả
1	Cookie [] getCookies(): Trả về mảng chứa tất cả các đối tượng Cookie mà client gửi với request này.
2	Enumeration getAttributeNames(): Trả về một Enumeration chứa tên của các thuộc tính có sẵn cho request này.
3	Enumeration getHeaderNames(): Trả về một Enumeration tất cả tên tiêu đề request này.
4	Enumeration getParameterName(): Trả về một Enumeration of String các đối tượng có chứa tên của các tham số chứa trong request này.

```
// Create cookies for first and last names.
      Cookie firstName = new Cookie("first_name",
            request.getParameter("first_name"));
      // Set expiry date after 24 Hrs for both the cookies.
      firstName.setMaxAge(60 * 60 * 24);
      // Add both the cookies in the response header.
      response.addCookie(firstName);
```

ServletRequest trong Java Servlet

TT	Phương pháp & Mô tả
5	HttpSession getSession(): Trả về phiên hiện tại được kết hợp với request này, hoặc nếu request không có phiên, hãy tạo một phiên.
6	HttpSession getSession (boolean): Trả về HttpSession hiện tại được liên kết với request này hoặc, nếu không có phiên hiện tại và giá trị của create là đúng, trả về một phiên mới.
7	Locale getLocale(): Trả về ngôn ngữ ưu tiên mà client sẽ chấp nhận nội dung, dựa trên tiêu đề Accept-Language.

```
// Create a session object if it is already not created.
      HttpSession session = request.getSession(true);
// Get session creation time.
      Date createTime = new Date(session.getCreationTime());
// Get last access time of this web page.
      Date lastAccessTime = new Date(session.getLastAccessedTime());
```

ServletRequest trong Java Servlet

TT	Phương pháp & Mô tả
8	getAttribute (String name): Trả về giá trị của thuộc tính được đặt tên như một đối tượng, hoặc null nếu không có thuộc tính của tên đã cho tồn tại.
9	ServletInputStream getInputStream(): Truy lục cơ thể của request dưới dạng dữ liệu nhị phân bằng ServletInputStream.
10	String getAuthType(): Trả về tên của lược đồ xác thực được sử dụng để bảo vệ servlet, ví dụ như "BASIC" hoặc "SSL" hoặc null nếu JSP không được bảo vệ.
11	String getCharacterEncoding(): Trả về tên của mã hoá ký tự được sử dụng trong phần thân request này.

Request trong Servlet

TT	Phương pháp & Mô tả
12	String getContentType(): Trả về kiểu MIME của phần thân request, hoặc null nếu loại không được biết.
13	String getContextPath(): Trả về phần của URI request chỉ ra ngữ cảnh của request.
14	String getHeader(String name): Trả về giá trị của tiêu đề request được chỉ định như là một Chuỗi.
15	String getMethod(): Trả về tên của phương thức HTTP mà request này được thực hiện, ví dụ như GET, POST, hoặc PUT.
16	String getParameter(String name): Trả về giá trị của tham số request như một String, hoặc null nếu tham số không tồn tại.

ServletRequest trong Java Servlet

TT	Phương pháp & Mô tả
17	String getPathInfo(): Trả về bất kỳ thông tin đường dẫn bổ sung liên quan đến URL mà client gửi khi thực hiện request này.
18	String getProtocol(): Trả về tên và phiên bản của giao thức theo request.
19	String getQueryString(): Trả về chuỗi truy vấn được chứa trong URL request sau đường dẫn.
20	String getRemoteAddr(): Trả về địa chỉ Giao thức Internet (IP) của client gửi request.
21	String getRemoteHost(): Trả về tên đầy đủ của client đã gửi request.
22	String getRemoteUser(): Trả về thông tin đăng nhập của người dùng request này, nếu người dùng đã được xác thực, hoặc null nếu người dùng không được xác thực.

ServletRequest trong Java Servlet

TT	Phương pháp & Mô tả
23	String getRequestURI(): Trả về một phần của URL của request này từ tên giao thức cho đến chuỗi truy vấn trong dòng đầu tiên của request HTTP.
24	String getRequestedSessionId(): Trả về ID phiên xác định bởi client.
25	String getServletPath(): Trả về một phần của URL của request này gọi là JSP.
26	String[] getParameterValues(String name): Trả về một mảng các đối tượng String chứa tất cả các giá trị mà tham số request cho trước có, hoặc null nếu tham số không tồn tại.
27	boolean isSecure(): Trả về một Boolean cho biết request này được thực hiện bằng một kênh an toàn, chẳng hạn như HTTPS.

ServletRequest trong Java Servlet

TT	Phương pháp & Mô tả
28	int getContentLength(): Trả về chiều dài, theo byte, của thân request và được cung cấp bởi luồng đầu vào, hoặc -1 nếu chiều dài không được biết.
29	int getIntHeader(String name): Trả về giá trị của tiêu đề request được chỉ định như là một int.
30	int getServerPort(): Trả về số cổng mà request này đã nhận được.

Ví dụ đọc HTTP Header với request trong servlet

- DisplayHeader.java

```
public void doGet(HttpServletRequest request, HttpServletResponse response)
        throws ServletException, IOException {
    // Set response content type
    response.setContentType("text/html");
    PrintWriter out = response.getWriter();
    String title = "Vi du doc thong tin HTTP Header";
    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"
            + "<table width = \"100%\" border = \"1\" align = \"center\">\n"
            + "<tr bgcolor = \"#949494\">\n"
            + "<th>Header Name</th><th>Header Value(s)</th>\n"
            + "</tr>\n");
```

Ví dụ đọc HTTP Header với request trong servlet

- DisplayHeader.java

```
// get header names
      @SuppressWarnings("rawtypes")
Enumeration headerNames = request.getHeaderNames();
      while (headerNames.hasMoreElements()) {
         String paramName = (String) headerNames.nextElement();
         out.print("<tr><td>" + paramName + "</td>\n");
         String paramValue = request.getHeader(paramName);
         out.println("<td> " + paramValue + "</td></tr>\n");
      }
      out.println("</table>\n</body></html>");
   }
}
```

Ví dụ đọc HTTP Header với request trong servlet

- Kết quả chạy trên Tomcat server

ServletResponse trong Java Servlet

- Khi Web server đáp ứng (**response**) yêu cầu của HTTP request. Một response thông thường bao gồm trạng thái (status), header, blank line, và document.

```
HTTP/1.1 200 OK
Content-Type: text/html
Header2: ...
...
HeaderN: ...
  (Blank Line)
<!doctype ...>
<html>
  <head>...</head>
  <body>
    ...
  </body>
</html>
```

Dòng trạng thái (status) bao gồm: phiên bản HTTP (trong ví dụ là HTTP / 1.1), mã trạng thái (trong ví dụ là 200) và một thông báo rất ngắn tương ứng với mã trạng thái (trong ví dụ là OK).

ServletResponse trong Servlet

- Các phương thức sau đây có thể được sử dụng để set HTTP Header response trong chương trình servlet. Các phương pháp này có sẵn trong đối tượng **HttpServletResponse**.

TT	Phương pháp & Mô tả
1	String encodeRedirectURL(String url): Mã hóa URL đã chỉ định để sử dụng trong phương thức sendRedirect hoặc, nếu mã hoá không cần thiết, sẽ trả về URL không thay đổi.
2	String encodeURL(String url): Mã hoá URL đã chỉ định bằng cách bao gồm ID phiên đó, hoặc, nếu mã hoá không cần thiết, trả về URL không thay đổi.
3	boolean containsHeader(String name): Trả về một Boolean cho biết header response đã được đặt chưa.

ServletResponse trong Servlet

TT	Phương pháp & Mô tả
4	boolean isCommitted(): Trả về một Boolean chỉ ra nếu đáp ứng đã được thực hiện.
5	void addCookie(Cookie cookie): Thêm cookie được chỉ định vào câu trả lời.
6	void addDateHeader(String name, long date): Thêm một header đáp ứng với tên và giá trị ngày.
7	void addHeader(String name, String value): Thêm một header đáp ứng với tên và giá trị đã cho.

ServletResponse trong Servlet

TT	Phương pháp & Mô tả
8	void addIntHeader(String name, int value): Thêm một header đáp ứng với tên và giá trị số nguyên.
9	void flushBuffer(): Buộc bất kỳ nội dung nào trong bộ đệm sẽ được ghi vào máy khách.
10	void reset(): Xóa bất kỳ dữ liệu nào tồn tại trong bộ đệm cũng như mã trạng thái và header.
11	void resetbuffer(): Xóa nội dung của bộ đệm cơ bản trong response mà không có header xoá hoặc mã trạng thái.

ServletResponse trong Servlet

TT	Phương pháp & Mô tả
12	void sendError(int sc): Gây response lỗi tới máy khách bằng cách sử dụng mã trạng thái được chỉ định và xóa bộ đệm.
13	void sendError(int sc, String msg): Gửi response lỗi đến máy khách sử dụng trạng thái được chỉ định.
14	void sendRedirect(String location): Gửi response chuyển hướng tạm thời tới khách hàng bằng cách sử dụng URL vị trí chuyển hướng được chỉ định.
15	void setBufferSize(int size): Thiết lập kích thước bộ đệm cho response.

ServletResponse trong Servlet

TT	Phương pháp & Mô tả
16	void setCharacterEncoding(String charset): Thiết lập mã hoá ký tự(ký tự MIME) của câu trả lời được gửi tới khách hàng, ví dụ như để UTF-8.
17	void setContentLength(int len): Đặt độ dài của nội dung trong response Trong HTTP servlet, phương pháp này đặt header HTTP Content-Length.
18	void setContentLength(int len): Thiết lập kiểu nội dung của câu trả lời được gửi đến khách hàng, nếu chưa trả lời.
19	void setContentType(String type): Thiết lập một header đáp ứng với tên và giá trị ngày.

```
response.setContentType("text/html");
response.setCharacterEncoding("UTF-8");
```

ServletResponse trong Servlet

TT	Phương pháp & Mô tả
20	void setHeader(String name, String value): Đặt header response với tên và giá trị đã cho.
21	void setIntHeader(String name, int value): Thiết lập header đáp ứng với tên và giá trị số đã cho.
22	void setLocale(Locale loc): Đặt vị trí của câu trả lời, nếu chưa trả lời. .
23	void setStatus(int sc): Đặt mã trạng thái cho response này.

Ví dụ ServletResponse trong servlet

- Phương thức setContentType() đã từng được sử dụng trong các ví dụ trước. Và nó cũng được sử dụng trong ví dụ này, ngoài ra chúng ta sử dụng phương thức **setIntHeader()** để set header là **Refresh**.

Refresh.java

```
// Method to handle GET method request.
public void doGet(HttpServletRequest request,
HttpServletResponse response)
        throws ServletException, IOException {

    // Set refresh, autoload time as 5 seconds
    response.setIntHeader("Refresh", 5);
    // Set response content type
    response.setContentType("text/html");
```

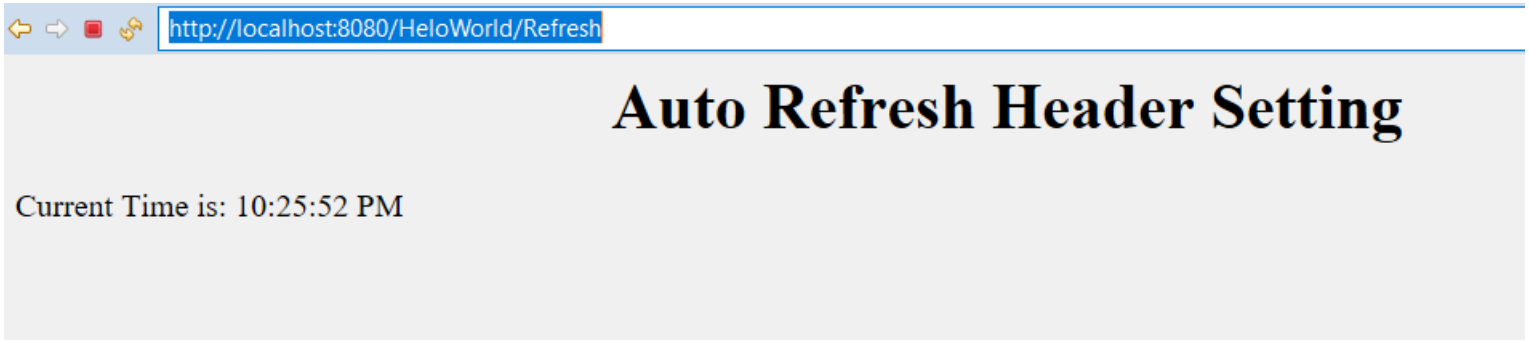
Ví dụ ServletResponse trong servlet


```
// Get current time
    Calendar calendar = new GregorianCalendar();
    String am_pm;
    int hour = calendar.get(Calendar.HOUR);
    int minute = calendar.get(Calendar.MINUTE);
    int second = calendar.get(Calendar.SECOND);

    if (calendar.get(Calendar.AM_PM) == 0)
       am_pm = "AM";
    else
       am_pm = "PM";

    String CT = hour + ":" + minute + ":" + second + " " + am_pm;
```

Ví dụ ServletResponse trong servlet


```
PrintWriter out = response.getWriter();
        String title = "Auto Refresh Header Setting";
        String docType = "<!doctype html">\n";
        out.println(docType + "<html>\n" +
              "<head><title>" + title + "</title></head>\n"+
              "<body bgcolor = \"#f0f0f0\">\n" +
              "<h1 align = \"center\">" + title + "</h1>\n" +
              "<p>Current Time is: " + CT + "</p>\n");
    }
}
```

Gọi servlet trên sẽ tạo ra kết quả sau: sẽ hiển thị thời gian hệ thống hiện tại và tự động **refresh** sau mỗi



TRƯỜNG ĐẠI HỌC
SƯ PHẠM KỸ THUẬT TP. HỒ CHÍ MINH
HCMC University of Technology and Education
HCMUTE

KHOA CÔNG NGHỆ THÔNG TIN
BỘ MÔN CÔNG NGHỆ PHẦN MỀM

LẬP TRÌNH WEB (WEBPR330479

Cookie trong Java Servlet

THS. NGUYỄN HỮU TRUNG

PR33 WEB 0479

Thông tin Giảng viên

- Ths. Nguyễn Hữu Trung
- Khoa Công Nghệ Thông Tin
- Trường Đại học Sư Phạm Kỹ Thuật TP.HCM
- 090.861.7108
- trungnh@hcmute.edu.vn
- https://www.youtube.com/@baigiai

Xử lý Cookie trong Servlet

- **Cookie** là các tập tin văn bản được lưu trữ trên client. Mục đích của cookie là để theo dõi các thông tin khác nhau. Ví dụ trường hợp remember login.
- Có ba bước liên quan đến xác định người dùng cũ quay trở lại hệ thống:
  - Tập lệnh của máy chủ gửi một tập hợp các cookie đến trình duyệt. Ví dụ tên, tuổi, hoặc số nhận dạng vv.
  - Trình duyệt lưu trữ thông tin này trên máy local để sử dụng trong tương lai.
  - Trong lần truy cập tiếp theo, trình duyệt gửi yêu cầu tới web server, nó sẽ gửi những thông tin cookie tới máy chủ và máy chủ sử dụng thông tin đó để xác định người dùng.

Ví dụ về một Cookie

- Cookie thường được đặt trong HTTP Header
  - HTTP/1.1 200 OK
  - Date: Fri, 13 Oct 2020 21:03:38 GMT
  - Server: Apache/1.3.9 (UNIX) PHP/4.0b3
  - Set-Cookie: name = abc; expires = Friday, 13-Oct-2020 22:03:38 GMT; path = /; domain = iotstar.vn
  - Connection: close
  - Content-Type: text/html

Một servlet sau đó sẽ có quyền truy cập cookie thông qua yêu cầu của phương thức request.getCookies() trả về một mảng các đối tượng Cookie.

Các phương thức xử lý cookie trong servlet

TT	Phương thức & Mô tả
1	public void setDomain(String pattern): thiết lập tên miền mà cookie áp dụng, ví dụ như iotstar.vn.
2	public String getDomain(): lấy tên miền mà cookie áp dụng, ví dụ như iotstar.vn.
3	public void setMaxAge(int expiry): Phương thức này đặt khoảng thời gian(tính bằng giây) trước khi cookie hết hạn. Nếu bạn không đặt điều này, cookie sẽ chỉ kéo dài cho phiên hiện tại.

Các phương thức xử lý cookie trong servlet

TT	Phương thức & Mô tả
4	public getMaxAge int(): Phương thức này trả về độ tuổi tối đa của cookie, được chỉ định bằng giây, Theo mặc định, -1 cho biết cookie sẽ tồn tại cho đến khi trình duyệt tắt máy.
5	public String getName(): Phương thức này trả về tên của cookie. Không thể thay đổi tên sau khi tạo.
6	public void setValue(String newValue): Phương thức này đặt giá trị kết hợp với cookie

Các phương thức xử lý cookie trong servlet

TT	Phương thức & Mô tả
7	public getValue String() : Phương thức này lấy giá trị kết hợp với cookie.
8	public void setPath(String uri): Phương thức này đặt đường dẫn đến cookie này được áp dụng. Nếu bạn không chỉ định đường dẫn, cookie sẽ được trả về cho tất cả các URL trong cùng thư mục với trang hiện tại cũng như tất cả các thư mục con.
9	public getPath String(): Phương thức này là đường dẫn đến cookie này được áp dụng.

Các phương thức xử lý cookie trong servlet

No.	Phương thức & Mô tả
10	public void setSecure(boolean flag): Phương thức này thiết lập giá trị boolean cho biết liệu cookie chỉ nên được gửi qua các kết nối được mật mã(tức là SSL).
11	public void setComment(String purpose): Phương thức này xác định một comment mô tả mục đích của một cookie. comment này hữu ích nếu trình duyệt trình bày cookie cho người dùng.
12	public String getComment(): Phương thức này trả về comment mô tả mục đích của cookie này, hoặc không hợp lệ nếu cookie không có comment.

Tạo Cookie trong Servlet

- Việc tạo cookie trong servlet bao gồm 3 bước sau:
  - **1. Tạo một đối tượng Cookie**: gọi constructor Cookie với các tham số tên và giá trị của cookie, cả hai đều có kiểu là String.

```
Cookie cookie = new Cookie("key","value");
```

Hãy ghi nhớ, tên và giá trị không nên chứa khoảng trắng hoặc bất kỳ ký tự nào sau đây: [ ] ( ) = , " / ? @ : ;

  - **2. Thiết định thời gian tồn tại cho Cookie**: sử dụng phương thức setMaxAge() để xác định cookie được sống trong thời gian bao lâu (tính bằng giây). Sau đây sẽ thiết lập một cookie sống trong 24 giờ.

```
cookie.setMaxAge(60 * 60 * 24);
```

  - **3. Đính kèm cookie vào HTTP response header**: sử dụng phương thức response.addCookie() để thêm các cookie và HTTP response header như sau: `response.addCookie(cookie);`

Ví dụ 1: xử lý cookie trong servlet

- Tạo Cookie : CreateCookie.java

```
//Nhận dữ liệu từ FORM
String ten = request.getParameter("ten");
String holot = request.getParameter("holot");

// Create cookies for first and last names.
 Cookie firstName = new Cookie("ten",ten);
 Cookie lastName = new Cookie("holot",holot);

 // Set expiry date after 24 Hrs for both the cookies.
 firstName.setMaxAge(60 * 60 * 24);
 lastName.setMaxAge(60 * 60 * 24);
      // Add both the cookies in the response header.
      response.addCookie(firstName);
      response.addCookie(lastName);

            PrintWriter out = response.getWriter();
     out.println("<b>First Name</b>: " + firstName.getValue()  + " - <b>Last Name</b>: " +
lastName.getValue());
```

```
<form action="createcookie"
method="GET">
Tên: <input type="text" name="ten"> <br />
 Họ lót: <input type="text" name="holot" />
            <input type="submit"
value="Submit" />
</form>
```

Index.html

Ví dụ 1: xử lý cookie trong servlet

- Đọc Cookie: ReadCookie.java

```
Cookie cookie = null;
Cookie[] cookies = null;
// Get an array of Cookies associated with this domain
cookies = request.getCookies();
// Set response content type
response.setContentType("text/html");
PrintWriter out = response.getWriter();

if (cookies != null) {
   out.println("<h2> Found Cookies Name and Value</h2>");
   for (int i = 0; i < cookies.length; i++) {
      cookie = cookies[i];
      out.print("Name : " + cookie.getName() + ",  ");
      out.print("Value: " + cookie.getValue() + " <br/>");
   }
} else {
   out.println("<h2>No cookies founds</h2>");
}
out.close();
```

Ví dụ 1: xử lý cookie trong servlet

- Xóa Cookie: Nếu bạn muốn xóa một cookie thì bạn chỉ cần làm theo ba bước sau:
  - Đọc một cookie hiện có và lưu trữ nó trong đối tượng Cookie.
  - Đặt tuổi cookie về **0** bằng cách sử dụng phương thức **setMaxAge()** để xóa cookie.
  - Add cookie trở lại response header.

DeleteCookie.java

```
Cookie cookie = null;
Cookie[] cookies = null;
// Get an array of Cookies associated with this domain
cookies = request.getCookies();
// Set response content type
response.setContentType("text/html");
PrintWriter out = response.getWriter();
if (cookies != null) {
   out.println("<h2> Cookies Name and Value</h2>");
   for (int i = 0; i < cookies.length; i++) {
      cookie = cookies[i];
      if ((cookie.getName()).compareTo("ten") == 0) {
         // delete cookie
         cookie.setMaxAge(0);
         response.addCookie(cookie);
out.print("Deleted cookie : " + cookie.getName() + "<br/>");
      }
      out.print("Name : " + cookie.getName() + ",  ");
      out.print("Value: " + cookie.getValue() + " <br/>");
   }
}
out.close();
```

Ví dụ 2: Cookie cho trang Login

- Bước 1: tạo trang Login.html

```
<form action="Login" method="Post">
Name: <input type="text" name="username "> <br />
 Pass: <input type="password" name="password" />
          <input type="submit" value="Login" />
</form>
```

- Bước 2: Tạo Class LoginServlet.java

```
@Override
protected void doPost(HttpServletRequest req, HttpServletResponse resp) throws ServletException,
IOException {
resp.setContentType("text/html");
//lấy dữ liệu từ tham số của form
String user = req.getParameter("username");
String pass = req.getParameter("password");
if(user.equals("trung") && pass.equals("123"))
{
Cookie cookie = new Cookie("username", user); //khởi tạo cookie
//thiết lập thời gian tồn tại 30s của cookie
cookie.setMaxAge(30);
//thêm cookie vào response
resp.addCookie(cookie);
//chuyển sang trang HelloServlet
resp.sendRedirect("/HelloServlet/hello");
}else {
//chuyển sang trang LoginServlet
resp.sendRedirect("/HelloServlet/login");
}}
```

Cấu hình URL: @WebServlet(urlPatterns= {"/login"})

Ví dụ 2: Cookie cho trang Login

Bước 3: Tạo trang HelloServlet.java

- Cấu hình URL: @WebServlet(urlPatterns= {"/hello","/xin-chao"})

```
@Override
protected void doGet(HttpServletRequest req, HttpServletResponse resp) throws
ServletException, IOException {
resp.setContentType("text/html");
PrintWriter printWriter = resp.getWriter();
String name="";
//Nhận cookie
Cookie[] cookie = req.getCookies();
for (Cookie c: cookie) {
        if(c.getName().equals("username")) {
                name = c.getValue();}}
        if(name.equals("")){
                //chuyển sang trang LoginServlet
                resp.sendRedirect("/HelloServlet/login");
        }
//hiển thị lên trang bằng đối tượng PrintWriter()
printWriter.println("Xin chao " + name);
}
```

Thực hiện chạy project để test kết quả

TRƯỜNG ĐẠI HỌC
SƯ PHẠM KỸ THUẬT TP. HỒ CHÍ MINH
HCMC University of Technology and Education
HCMUTE

KHOA CÔNG NGHỆ THÔNG TIN
BỘ MÔN CÔNG NGHỆ PHẦN MỀM

LẬP TRÌNH WEB (WEBPR330479

HttpSession trong Java Servlet

THS. NGUYỄN HỮU TRUNG

PR33
WEB
0479

Session trong Servlet

- **HTTP** là một giao thức "stateless" nghĩa là mỗi lần client truy xuất một trang Web, client sẽ mở một kết nối riêng đến máy chủ Web và máy chủ sẽ tự động **không giữ lại bất kỳ thông tin nào** về yêu cầu của client trước đó.
- Có ba cách sau để thực hiện theo dõi phiên (**session tracking**) giữa client và máy chủ web:
  - Sử dụng Cookies
  - Sử dụng Hidden Fields của HTML From
  - Sử dụng URL Rewriting
- Ngoài ra Servlet cung cấp HttpSession để xác định người dùng ghé thăm trang web và lưu trữ thông tin về người dùng đó

Đối tượng HttpSession

- Tạo đối tượng HttpSession bằng cách gọi phương thức getSession() của HttpServletRequest, như sau:
  - HttpSession session = request.getSession();
  - Bạn cần gọi request.getSession() trước khi gửi bất kỳ nội dung nào đến client. Dưới đây là tóm tắt các phương thức quan trọng hiện có của đối tượng HttpSession.

Đối tượng HttpSession

TT	Phương thức & Mô tả
1	**public Object getAttribute(String name)**: trả về đối tượng bị ràng buộc với tên được chỉ định trong session này, hoặc null nếu không có đối tượng nào bị ràng buộc dưới tên.
2	**public Enumeration getAttributeNames():** trả về một Enumeration of String các đối tượng có chứa tên của tất cả các đối tượng ràng buộc vào session này.

Đối tượng HttpSession

TT	Phương thức & Mô tả
3	**public long getCreationTime():** trả về thời gian khi session này được tạo ra, được đo bằng mili giây kể từ nửa đêm ngày 1 tháng 1 năm 1970 GMT.
4	**public String getId():** trả về một chuỗi chứa mã định danh duy nhất được gán cho session này.
5	**public long getLastAccessedTime():** trả về thời gian truy cập cuối cùng của session, theo định dạng mili giây kể từ nửa đêm ngày 1 tháng 1 năm 1970 GMT

Đối tượng HttpSession

TT	Phương thức & Mô tả
6	**public int getMaxInactiveInterval():** trả về khoảng thời gian tối đa (giây), rằng vùng chứa servlet sẽ giữ session mở giữa các lần truy cập của client.
7	**public void invalidate():** làm mất hiệu lực session này và unbinds bất kỳ đối tượng liên quan đến nó.
8	**public boolean isNew():** trả về true nếu client chưa biết về session hoặc nếu client không tham gia session làm việc.

Đối tượng HttpSession

TT	Phương thức & Mô tả
9	**public void removeAttribute(String name):** loại bỏ các đối tượng bị ràng buộc với tên quy định từ session này.
10	**public void setAttribute(String name, Object value):** liên kết một đối tượng đến session này, sử dụng tên được chỉ định.
11	**public void setMaxInactiveInterval(int interval):** chỉ định thời gian, tính bằng giây, giữa các yêu cầu của client trước khi bộ chứa servlet sẽ làm mất hiệu lực session này.

Ví dụ 1: Session trong Servlet

- CreateSession.java

```
@Override
protected void doGet(HttpServletRequest req, HttpServletResponse resp)
throws ServletException, IOException {
        //khởi tạo session
        HttpSession s = req.getSession();
        //Gán dữ liệu vào session
        s.setAttribute("ten", "Nguyễn Hữu Trung");
        s.setAttribute("tuoi", new Integer(40));
        //thiết lập thời gian tồn tại session
        s.setMaxInactiveInterval(30);
        //hiển thị thông báo lên web
        resp.setContentType("text/html");
        resp.setCharacterEncoding("UTF-8");
        PrintWriter out = resp.getWriter();
        out.println("Xin chào bạn session đã được tạo");
        out.close();
}
```

Ví dụ 1: Session trong Servlet

- ShowSession.java

```
@Override
protected void doGet(HttpServletRequest req, HttpServletResponse resp) throws ServletException,
IOException {
// hiển thị session lên web
resp.setContentType("text/html");
resp.setCharacterEncoding("UTF-8");
PrintWriter out = resp.getWriter();
String ten = "";
HttpSession s = req.getSession(); // khởi tạo session
Object obj= s.getAttribute("ten"); // truy xuất dữ liệu từ session
//kiểm tra đối tượng Object có null không
if(obj != null) {
        ten = String.valueOf(obj); //ép kiểu về String
}else {
        resp.sendRedirect("/HelloServlet/creatsession"); //nếu null thì chuyển về trang tạo session
}
int tuoi = (Integer)s.getAttribute("tuoi");//ép kiểu
//Hiển thị session lên web
out.println("Xin chào bạn: " + ten + " tuổi: " + tuoi);
out.close();
}
```

Xóa Session Data

- Bạn có thể xóa session data với các tùy chọn như sau:
  - **Xoá một thuộc tính cụ thể**: bạn có thể gọi phương thức public void removeAttribute(String name) để xóa giá trị kết hợp với một khoá cụ thể.
  - **Xóa toàn bộ sesssion**: bạn có thể gọi phương thức public void invalidate() để xóa toàn bộ sesssion.
  - **Cài đặt thời gian chờ cho session**: bạn có thể gọi phương thức public void setMaxInactiveInterval(int interval)
  - **Đăng xuất user**: các máy chủ hỗ trợ servlet 2.4, bạn đăng xuất khách hàng ra khỏi Web server và làm mất hiệu lực tất cả session của tất cả người dùng.
  - **Cấu hình web.xml**: nếu bạn đang sử dụng Tomcat (30 phút), ngoài các phương pháp đã đề cập ở trên, bạn có thể thiết định thời gian session trong tệp web.xml như sau:

```
<session-config>
   <session-timeout>15 phút (900 giây)</session-timeout>
</session-config>
```

Cách hoạt động

- Cookie
  - Set ID vào cookie để nhận dạng
- Hidden Field trong Form
  - Set ID vào một trường ẩn trong Form
  - <input type="hidden" name ="sessionid", value="123">
- URL Rewriting
  - http://iotstar.vn/home;sessionid=123

HttpSession

- Là một interface trong Servlet giúp lưu các thông tin người dùng servlet khác nhau
- HttpSession session = request.getSession();

```
//Thiết lập Session
HttpSession session = req.getSession();
session.setAttribute("uname", “Nguyễn Hữu Trung”);
session.setMaxInactiveInterval(10); //Thiết lập thời gian tồn tại Session
//Lấy thông tin Session
String uname="";
Object obj = session.getAttribute("uname");
if (obj != null) {
uname = String.valueOf(obj);
//hiển thị lên trang bằng đối tượng PrintWriter()
printWriter.println(uname);
}else {
//chuyển sang trang LoginServlet
resp.sendRedirect("/HelloServlet/login");
}
```

Login bằng session


```
<form action="login" method="post">
UserName:<input type="text" name="username"><br/>
Password:<input type="password" name="password"><br/>
<input type="submit" value="login">
</form>
```

Login.html

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

if (username.equals("trungnh")&& password.equals("123")) {
out.print("Chào mừng bạn, " + username);
HttpSession session = request.getSession();
session.setAttribute("name", username);
} else {
out.print("Tài khoản hoặc mật khẩu không chính xác");
request.getRequestDispatcher("Login.html").include(request,
response);
}
```

LoginServlet.java

Logout.java


```
//Hủy Session
HttpSession session = request.getSession();
session.invalidate();
request.getRequestDispatcher("Login.html").include(request,
response);
```

```
if(session!=null){
String name=(String)session.getAttribute("name");
out.print("Chào bạn, "+name+" đến với trang quản lý tài khoản");
}
else {
out.print("Xin vui lòng đăng nhập");
resp.sendRedirect("/HelloServlet/Login.html");
}
```

Profile.java

Bài tập rèn luyện

- Yêu cầu: Thực hiện chức năng login có lưu thông tin của mình bằng Session. Nộp video có thuyết minh quá trình làm cho giảng viên.

TRƯỜNG ĐẠI HỌC
SƯ PHẠM KỸ THUẬT TP. HỒ CHÍ MINH
HCMC University of Technology and Education
HCMUTE

KHOA CÔNG NGHỆ THÔNG TIN
BỘ MÔN CÔNG NGHỆ PHẦN MỀM

LẬP TRÌNH WEB (WEBPR330479

Page Redirect, RequestDispatcher trong Java Servlet

THS. NGUYỄN HỮU TRUNG

PR33
WEB
0479

Page Redirect trong Servlet

- **Chuyển hướng trang** (page redirect) là một kỹ thuật mà client được chuyển đến một trang mới ngoài trang được request. Chuyển hướng trang thường được sử dụng khi tài liệu chuyển đến vị trí mới hoặc có thể là do cân bằng tải(load balancing).
- Cách đơn giản nhất để thực hiện page redirect trong servlet là sử dụng phương thức sendRedirect() của đối tượng response. Khai báo của phương thức này như sau:

```
public void HttpServletResponse.sendRedirect(String location)
    throws IOException
```

- PageRedirect.java

```
public void doGet(HttpServletRequest request,
HttpServletResponse response)
        throws ServletException, IOException {
    // Set response content type
    response.setContentType("text/html");
    // New location to be redirected
    String site = new String("http://iotstar.vn");
        response.sendRedirect(site);
}
```

RequestDispatcher trong Servlet

- Phân phối request tới các nguồn tài nguyên khác
- 02 phương thức chính là forward() và include()

```
RequestDispatcher rd = req.getRequestDispatcher("/index.jsp");
rd.forward(req, resp);
```

```
RequestDispatcher rd = req.getRequestDispatcher("/home");
rd.forward(req, resp);
```

```
RequestDispatcher rd = req.getRequestDispatcher("/index.jsp");
rd.include(req, resp);
```

```
RequestDispatcher rd = req.getRequestDispatcher("/home");
rd.include(req, resp);
```

RequestDispatcher và SendRedirect

- **SendRedirect ():**
  - Phương thức này được khai báo trong HttpServletResponse nó là một **Interface**.
  - **Cách sử dụng** : void *sendRedirect(String url)*
  - chuyển hướng đến tài nguyên khác như tên miền khác hoặc các đường dẫn là các máy chủ khác nhau để xử lý.
  - thanh địa chỉ chúng tôi có thể thấy địa chỉ chuyển hướng mới.
  - muốn gữi dữ liệu đi thì phải lưu trữ các dữ liệu trong phiên làm việc hoặc đi cùng với URL.
- **Forward():**
  - Phương thức này được khai báo trong RequestDispatcher**,** nó là **Interface.**
    Cách sử dụng : *forward*(ServletRequest request, ServletResponse response)
  - Tốc độ chuyên tiếp đến trang xử lý là nhanh hơn.
  - Gửi dữ liệu sang một trang được chuyển tiếp đến bằng cách sử dụng request.setAttribute ()
  - không thể thấy địa chỉ chuyển tiếp như vậy sẽ bảo mật hơn
  - chuyển đến tài nguyên khác **trong cùng một máy chủ** để tiếp tục xử lý

```
RequestDispatcher rd = req.getRequestDispatcher("/index.jsp");
rd.forward(req, resp);
```

```
//chuyển sang trang LoginServlet
resp.sendRedirect("/HelloServlet/login");
```

TRƯỜNG ĐẠI HỌC
SƯ PHẠM KỸ THUẬT TP. HỒ CHÍ MINH
HCMC University of Technology and Education
HCMUTE

KHOA CÔNG NGHỆ THÔNG TIN
BỘ MÔN CÔNG NGHỆ PHẦN MỀM

LẬP TRÌNH WEB (WEBPR330479

Error Handler trong Servlet

THS. NGUYỄN HỮU TRUNG

PR33
WEB
0479

Error Handler trong Servlet

- Bước 1: Tạo class ErrorHandleServlet

```
@WebServlet(urlPatterns= {"/error"})

@Override
protected void doGet(HttpServletRequest req, HttpServletResponse
resp) throws ServletException, IOException {
    resp.setContentType("text/html");
    resp.setCharacterEncoding("UTF-8");
    PrintWriter printWriter = resp.getWriter();
    String name="Xảy ra lỗi, trang không tồn tại“;
    //hiển thị lên trang bằng đối tượng PrintWriter()
    printWriter.println(name);
}

@Override
protected void doPost(HttpServletRequest req, HttpServletResponse resp) throws
ServletException, IOException {
        doGet(req, resp);
}
```

Error Handler trong Servlet

Bước 2: Chỉnh thông tin trong web.xml

```
<error-page>
 <error-code>404</error-code>
 <location>/error</location>
 </error-page>

 <error-page>
 <error-code>500</error-code>
 <location>/error</location>
 </error-page>


 <error-page>
 <exception-type>java.io.IOException</exception-type>
 <location>/error</location>
 </error-page>
```

Kiểm tra status_code để chuyển trang lỗi


```
@Override
protected void doGet(HttpServletRequest req, HttpServletResponse resp) throws ServletException, IOException {

 Throwable throwable = (Throwable) req.getAttribute("javax.servlet.error.exception");
    Integer statusCode = (Integer) req.getAttribute("javax.servlet.error.status_code");
    String servletName = (String) req.getAttribute("javax.servlet.error.servlet_name");

    if (servletName == null) {
      servletName = "Unknown";
    }
    String requestUri = (String)req.getAttribute("javax.servlet.error.request_uri");

    if (requestUri == null) {
      requestUri = "Unknown";
    }

if(statusCode == 404) {
resp.sendRedirect("/HelloServlet/loi404.jsp");

}
else if(statusCode == 500) {

resp.sendRedirect("/HelloServlet/loi500.jsp");

}else {

}}
```

Deploy project lên Server

- Kích phải vào project -> Export -> tìm war file
- Chọn thư mục chứa war file: trong thư mục webapps của tomcat.
- Mở CMD truy cập vào thư mục bin của tomcat
- Gõ startup.bat để khởi động tomcat
- Mở trình duyệt lên và truy xuất đường dẫn web
- http://localhost:8080/HelloServlet