

**TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN
KHOA CÔNG NGHỆ THÔNG TIN**

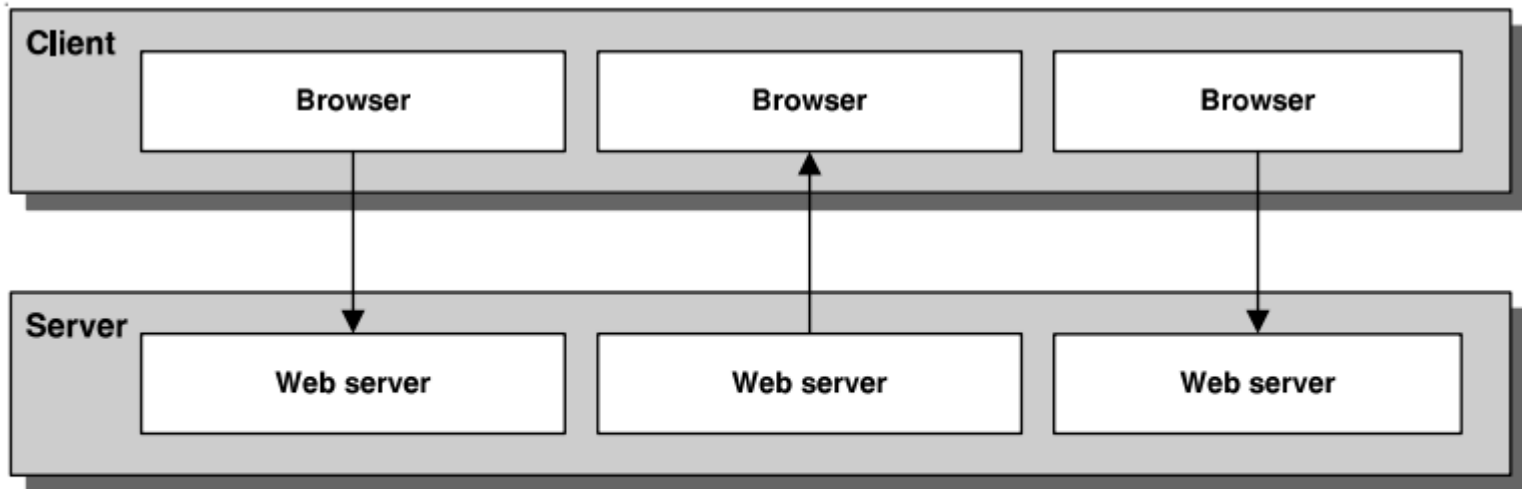
MVC 2 - SESSION & COOKIE

Nguyễn Hoàng Anh
nhanh@fit.hcmus.edu.vn

Nội dung

- Session Tracking
 - Session API
 - URL Encoding
 - Hidden Form Field
- Cookies
- MVC 2 & Session Tracking

Session

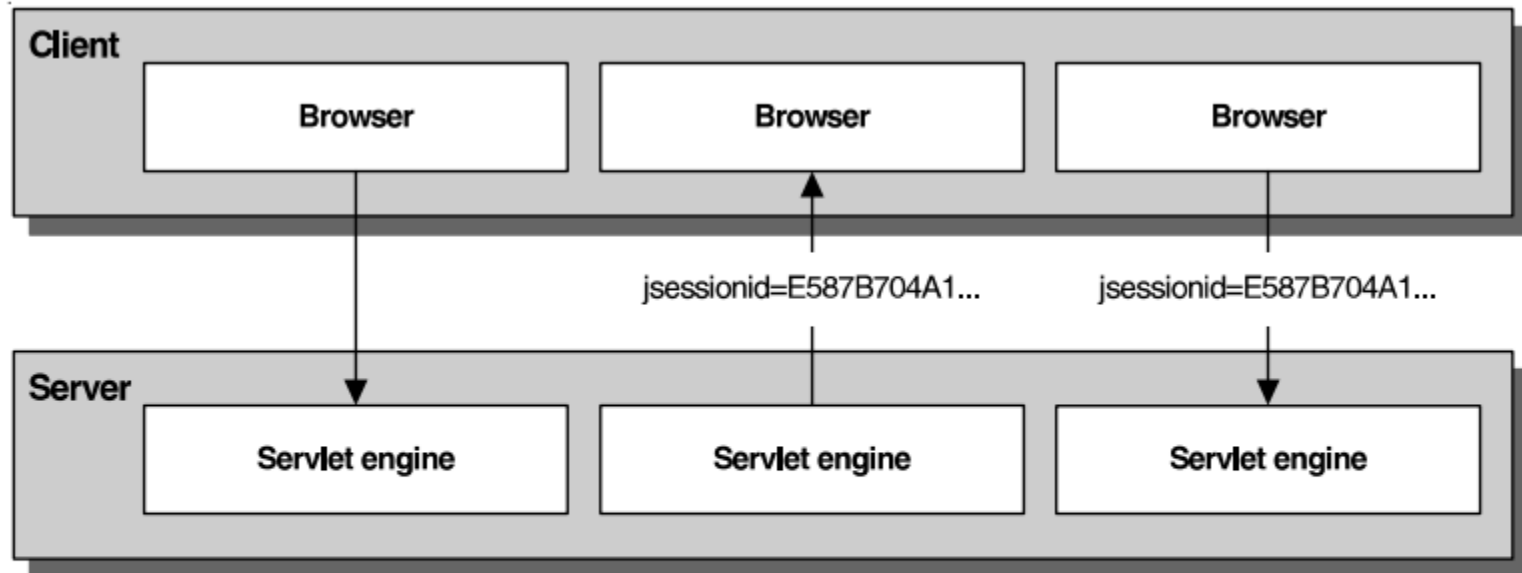


First HTTP Request:
The browser requests a page.

First HTTP Response:
The server returns the requested page and drops the connection.

Following HTTP Requests:
The browser requests a page. The web server has no way to associate the browser with its previous request.

Session



First HTTP Request:

The browser requests a JSP or servlet. The servlet engine creates a session object and assigns an ID for the session.


First HTTP Response:

The server returns the requested page and the ID for the session.

Following HTTP Requests:

The browser requests a JSP or servlet. The servlet engine uses the session ID to associate the browser with its session object.

Session





hoanganhis, open the Amazon.com Store Card and **Get \$10 Off Instantly**

Your current subtotal: \$ 5,218.99 [Apply now](#)

Gift Card savings: - \$ 10.00

Your cost after savings: \$ 5,208.99

Shopping Cart

	Price	Quantity
 <p>4-Year Laptop/Netbook Accident Protection Plan (\$2000-2500) by Smartfood</p> <p>In Stock</p> <p>Gift options not available. Learn more</p> <p>Delete Save for later</p>	\$320.99	2
 <p>Apple MacBook Pro MGXC2LL/A 15.4-Inch Laptop with Retina Display (NEWEST VERSION) by Apple Computer</p> <p>In Stock</p> <p>Gift options not available. Learn more</p> <p>Delete Save for later</p>	\$2,449.00 You save: \$50.00 (2%)	2
Subtotal (4 items): \$5,539.98		

Subtotal (4 items): \$5,539.98


[Proceed to checkout](#)

or


[Sign in](#) to turn on 1-Click ordering.

Estimate your shipping and tax [▼](#)

Frequently Bought With 4-Year Laptop/Netbook Accident P...



ASUS ROG G750JZ-DS71...
★★★★☆ (73)
~~\$2,499.00~~ **\$2,111.71**
[Add to Cart](#)



Apple MacBook Pro MGX...
★★★★★ (108)
~~\$2,499.00~~ **\$2,449.00**
[Add to Cart](#)

Session

- HTTP là stateless protocol. Để duy trì state, web application phải sử dụng session tracking.
- Mặc định, Servlet sử dụng cookie để lưu trữ session id cho mỗi browser. Mỗi lần request, browser gửi cookie đến server.
- Khi người dùng tắt cookie tại browser, Servlet có thể sử dụng URL Encoding để lưu trữ session id trong URL cho mỗi trang web.
- Để lưu trữ dữ liệu cho mỗi session, server tạo một đối tượng session.

Session – Các phương thức chính

- Get Session

- `HttpSession session = request.getSession();`

- Set Attribute

- `session.setAttribute(String key, Object value)`

- Get Attribute

- `Object obj = session.getAttribute(String key)`

- Remove Attribute

- `session.removeAttribute(String name)`

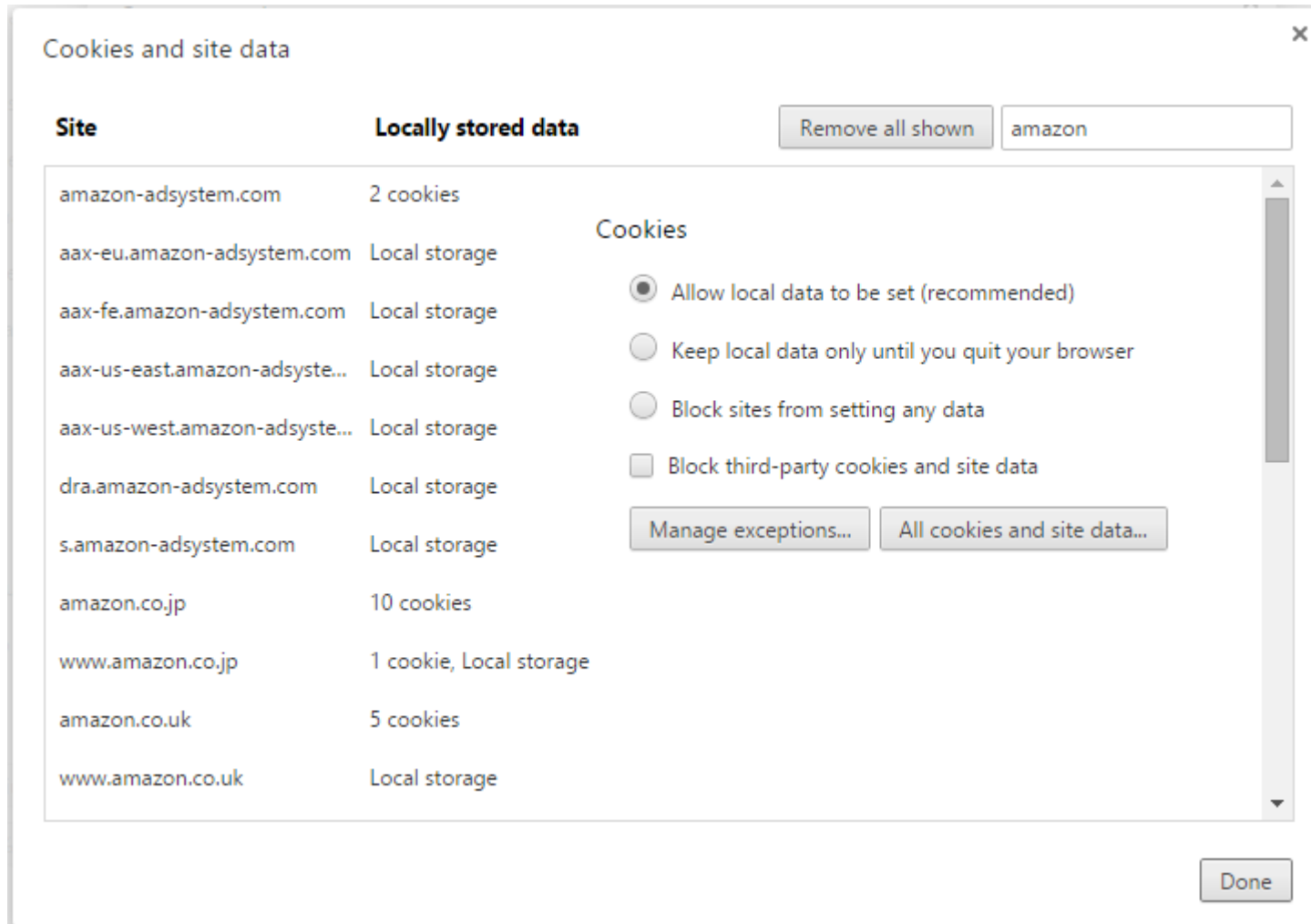
Session – Một số phương thức khác

- `Enumeration` enum = session.`getAttributeNames()`
- `String` id = session.`getId()`
- `boolean` new = session.`isNew()`
- session.`setMaxInactiveInterval`(int seconds)
- session.`invalidate()`


Cookies

- **Loại 1:** *a per-session cookie* được lưu trữ trên browser đến khi người dùng đóng browser.
- **Loại 2:** *persistent cookie* có thể lưu trữ trên ổ cứng người dùng đến 3 năm.

Cookies – Chrome



Cookies – Chrome



hoanganhis, open the Amazon.com Store Card and **Get \$10 Off Instantly**



Your current subtotal: \$ 5,218.99 [Apply now](#)

Gift Card savings: - \$ 10.00

Your cost after savings: \$ 5,208.99

*10 Instant Gift Card

Shopping Cart

	Price	Quantity
 <p>4-Year Laptop/Netbook Accident Protection Plan (\$2000-2500) by Smartfood</p> <p>In Stock</p> <p>Gift options not available. Learn more</p> <p>Delete Save for later</p>	\$320.99	2
 <p>Apple MacBook Pro MGXC2LL/A 15.4-Inch Laptop with Retina Display (NEWEST VERSION) by Apple Computer</p> <p>In Stock</p> <p>Gift options not available. Learn more</p> <p>Delete Save for later</p>	\$2,449.00 You save: \$50.00 (2%)	2
Subtotal (4 items): \$5,539.98		

Subtotal (4 items): \$5,539.98


[Proceed to checkout](#)

or

[Sign in](#) to turn on 1-Click ordering.

Estimate your shipping and tax [▼](#)

Frequently Bought With 4-Year Laptop/Netbook Accident P...




ASUS ROG G750JZ-DS71...

★★★★★ (73)

~~\$2,499.00~~ **\$2,111.71**

[Add to Cart](#)



Apple MacBook Pro MGX...

★★★★★ (108)

~~\$2,499.00~~ **\$2,449.00**

[Add to Cart](#)

Cookies – Chrome

Cookies

- ☐ Allow local data to be set (recommended)
- ☐ Keep local data only until you quit your browser
- ☒ Block sites from setting any data
- ☐ Block third-party cookies and site data

Manage exceptions...

All cookies and site data...

amazon
Try Prime

Your Amazon.com

Today's Deals

Gift Cards

Sell

Help

Shop by
Department ▾

Search

All ▾

Go

Hello, Sign in
Your Account ▾

Try
Prime ▾

0
Cart ▾

Wish
List ▾

ALL-NEW

fire HD



STARTING AT \$99
[> Learn more](#)



Important messages about items in your Cart:

Please Enable Cookies in your Web Browser to Continue.

[Learn more](#) about cookies and how to enable them.

Once you have enabled cookies, please click [here to continue shopping](#).

Session – URL Encoding

- Khi sử dụng **URL Encoding** phải encode tất cả các URL trong ứng dụng web.

- Phương thức sử dụng: `encodeURL(String url)`

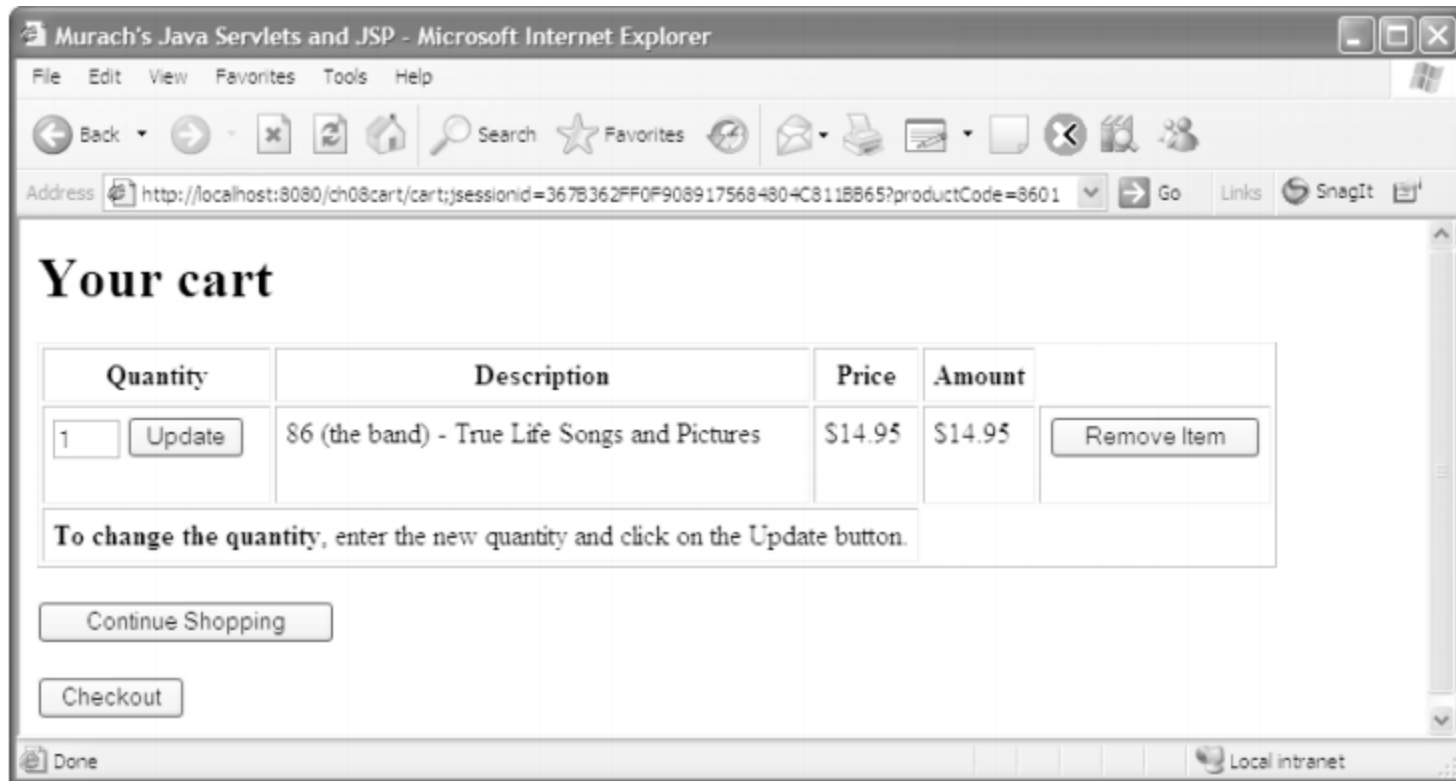
- Encode URL trong **Form Tag**

```
<form action="<%=response.encodeURL("cart")%>"  
        method="post"> . . . </form>
```

- Encode URL trong **A Tag**

```
<a href="<%=response.encodeURL("cart?id=86")%>"  
    Add to cart </a>
```

Session – URL Encoding



Session – Thread safe

- Mỗi servlet tạo một đối tượng session cho nhiều request từ một client.
- Nếu client chỉ có một cửa sổ của browser được mở, việc truy xuất đến đối tượng session là thread-safe.
- Nếu client có nhiều cửa sổ của browser được mở, việc truy xuất đến đối tượng session là không thread-safe.

Session – Thread safe

```
synchronized(session){  
    session.setAttribute("cart", cart);  
}
```

```
synchronized(session){  
    Cart cart = (Cart) session.getAttribute("cart");  
}
```


Cookies

- **Loại 1:** *a per-session cookie* được lưu trữ trên browser đến khi người dùng đóng browser.
- **Loại 2:** *persistent cookie* có thể lưu trữ trên ổ cứng người dùng đến 3 năm.

Cookies

- Cookie là 1 cặp name/value được lưu trữ tại browser.
- Trên server, server tạo cookie và gửi xuống client lưu trữ.
- Tại client, browser lưu trữ cookie trên máy và sẽ gửi ngược lại server cùng với các request.
- Cookie có thể được lưu trữ đến 3 năm tại client.
- Một số ví dụ
 - `jsessionid=E85FAC04E331FFCA55549B10B7C7A4FA`
 - `userid=nhanh`
 - `email=nhanh@fit.hcmus.edu.vn`
 - `password=nhanh-us`

Cookies - Một số trường hợp thông dụng

- Cho phép người dùng bỏ qua form đăng nhập, form đăng ký.
- Tùy chỉnh trang web.
- Hiện thị quảng cáo phù hợp.

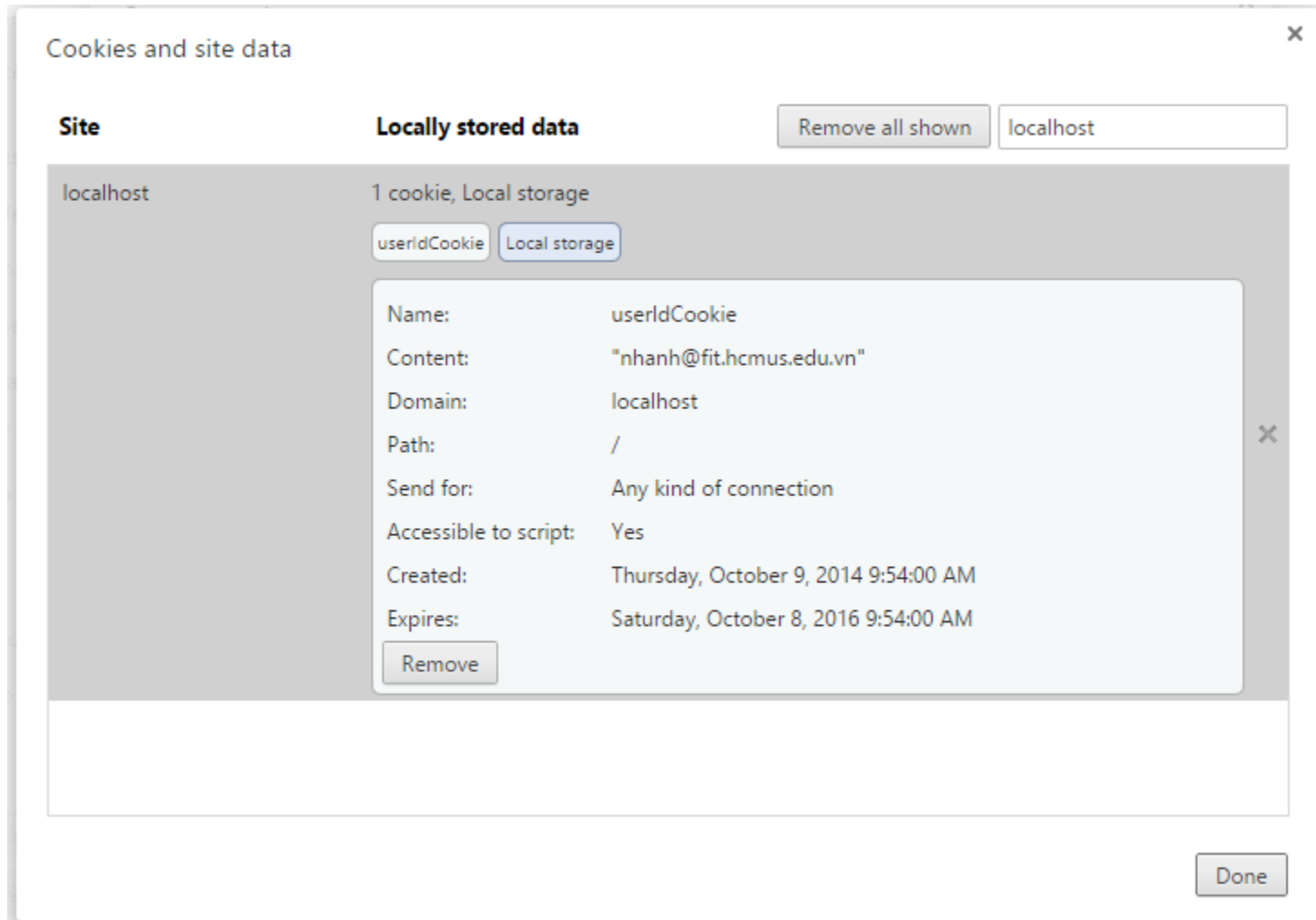
Cookie – Một số phương thức thường dùng

- Tạo cookie:
 - `Cookie cookie = new Cookie(String n, String v)`
- Sử dụng cookie
 - `cookie.setMaxAge(int maxAgeInSeconds)`
 - `cookie.setPath(String path)`
 - `cookie.getName()`
 - `cookie.getValue()`
- Gửi cookie về browser cùng với response
 - `response.addCookie(cookie)`

Cookie – Tạo và gửi về Client

```
String user="nhanh@fit.hcmus.edu.vn";  
Cookie userIdCookie = new Cookie("userIdCookie", user);  
userIdCookie.setMaxAge(60*60*24*365*2);  
userIdCookie.setPath("/");  
response.addCookie(userIdCookie);
```

Cookie



Cookie – Lấy thông tin Cookie

```
Cookie [] cookies = request.getCookies();
String cookieName = "userIdCookie";
String cookieValue = "";
for(int i=0; i< cookies.length; i++){
    Cookie cookie = cookies[i];
    if(cookieName.equals(cookie.getName())){
        cookieValue = cookie.getValue();
    }
}
```

Cookie – Huỷ Cookie

```
Cookie [] cookies = request.getCookies();  
for(int i=0; i< cookies.length; i++){  
    Cookie cookie = cookies[i];  
    cookie.setMaxAge(0);  
    cookie.setPath("/");  
    response.addCookie(cookie);  
}
```


URL Rewriting & Hidden Fields

- URL Rewriting được sử dụng để truyền các parameter từ client lên server.
 - Thêm các parameter vào URL.
- Ví dụ
 - `url?param1=value1¶m2=value2&...`
- Giới hạn
 - Các browser giới hạn số lượng ký tự được truyền theo url (2000 ký tự).
 - Khó xử lý các ký tự khoảng trắng và các ký tự đặc biệt.

URL Rewriting & Hidden Fields

- Hidden Form Field được sử dụng để truyền dữ liệu từ browser lên server.

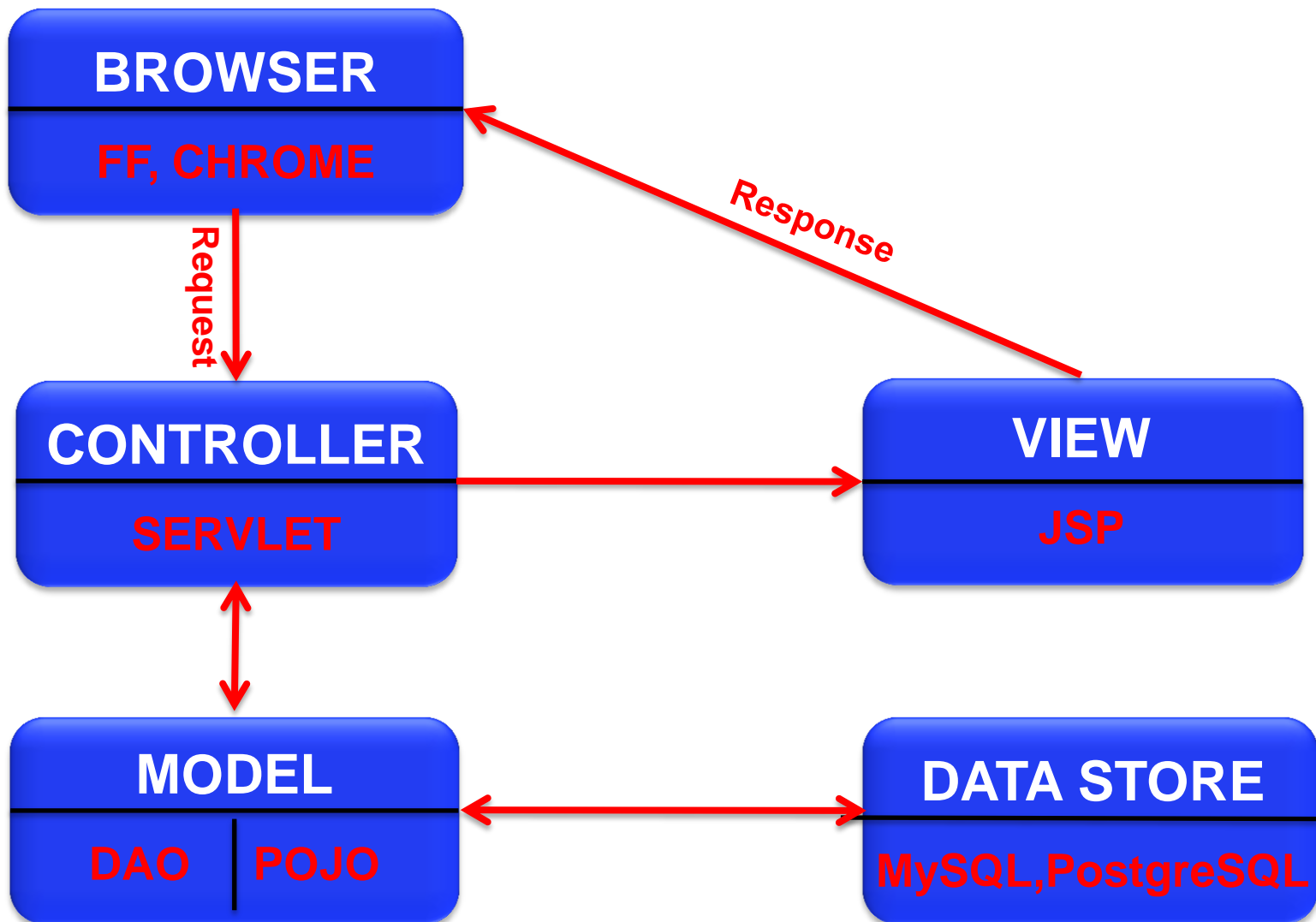
```
<form action="cart" method="post">  
  <input type="submit"  
    value="Add to cart"/>  
  <input type="hidden"  
    name="productCode" value="8601"/>  
</form>
```

86 (the band) - True Life Songs and Pictures	\$14.95	<input type="button" value="Add To Cart"/>
--	---------	--

URL Rewriting & Hidden Fields

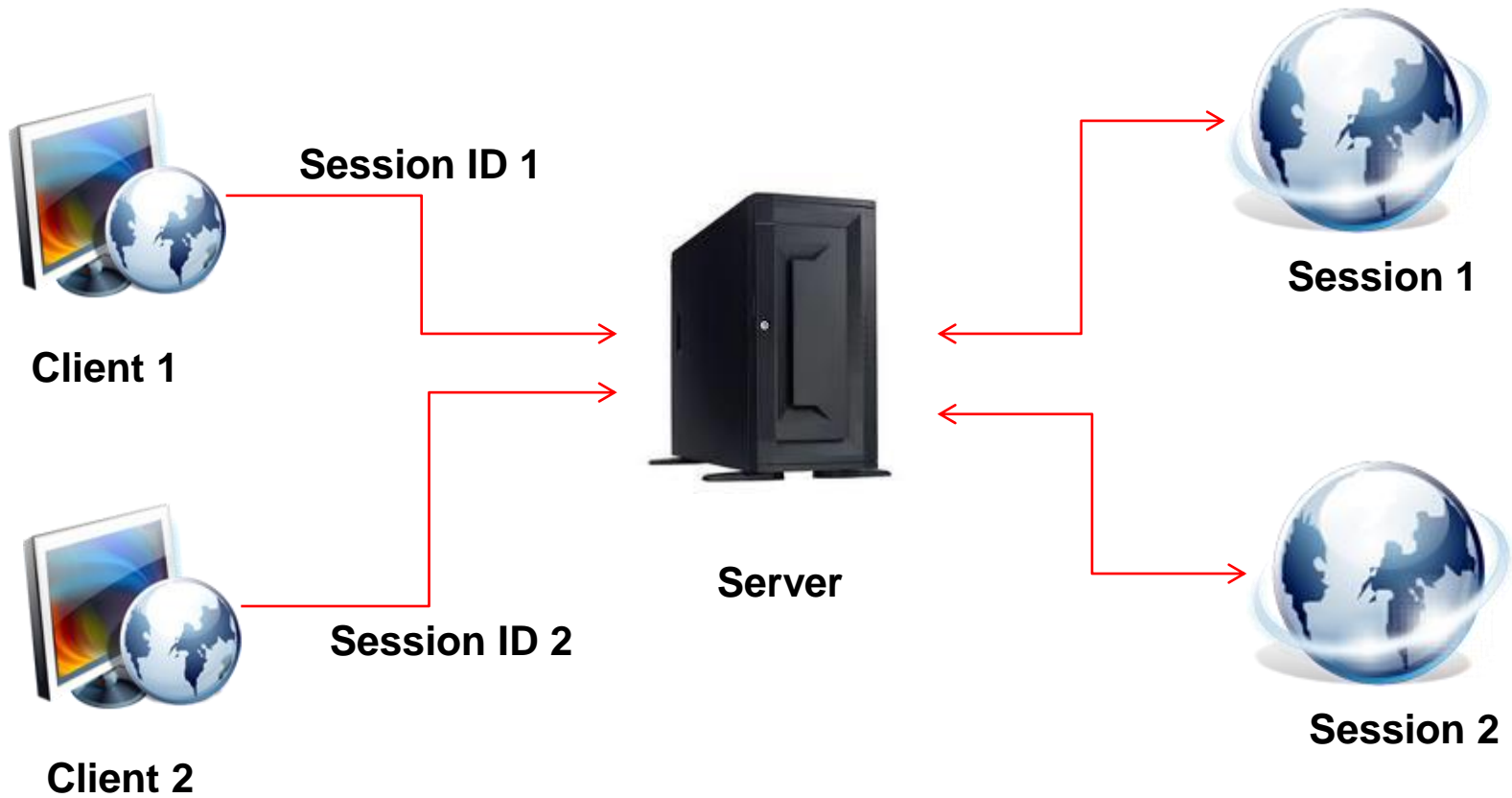
- **Giới hạn** của **Hidden Fields** là người dùng ở client có thể **xem các parameter** bằng cách *view source* .

MVC 2



Controller (Servlet) → **SESSION** → View (JSP)

- Session



Controller (Servlet) → **SESSION** → View (JSP)

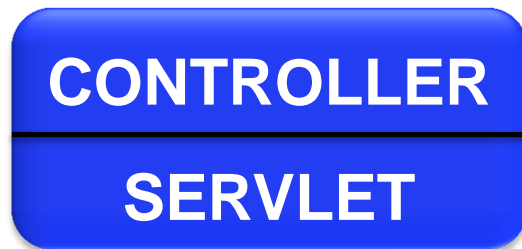
- **Forward**

- Người dùng không nhìn thấy URL thay đổi
- Người dùng không thể Bookmark
- Quá trình chuyển dữ liệu **Servlet → JSP**

- **SendRedirect**

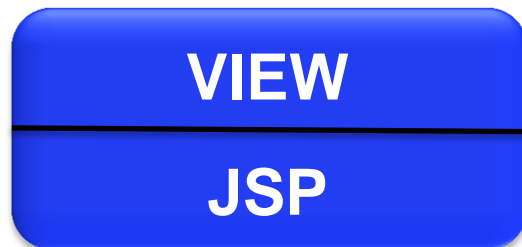
- Người dùng nhìn thấy URL thay đổi
- Người dùng có thể Bookmark trang JSP
- Quá trình chuyển dữ liệu **JSP → Servlet → JSP**
- Bên View cần kiểm tra dữ liệu đã tồn tại chưa

Controller (Servlet) → **SESSION** → View (JSP)



```
Object value1 = ...  
Object value2 = ...  
HttpSession session = request.getSession();  
session.setAttribute ("Key1", value)  
session.setAttribute ("Key2", value)  
...
```

Forward



```
Object value1 = session.getAttribute ("Key1")  
Object value2 = session.getAttribute ("Key2")  
...
```

Controller (Servlet) → SESSION → View (JSP)

- Controller (Servlet)

POJO pojo = . . .

```
HttpSession session = request.getSession();
```

```
session.setAttribute("key", pojo);
```

```
String url="url view";
```

```
RequestDispatcher rd =
```

```
    request.getRequestDispatcher(url);
```

```
rd.forward(request, response);
```


Controller (Servlet) → **SESSION** → View (JSP)

- View (JSP)
 - Java Bean

```
<jsp:useBean id="key"  
              type="POJO" scope="session" />
```

. . .

- Scriptlet

```
<%
```

```
POJO pojo=(POJO)session.getAttribute("Key");
```

```
%>
```

Controller (Servlet) → SESSION → View (JSP)

- Controller (Servlet)

```
ArrayList<POJO> ds = . . .
```

```
HttpSession session = request.getSession();
```

```
session.setAttribute("key", ds);
```

```
String url="url view";
```

```
RequestDispatcher rd =
```

```
    request.getRequestDispatcher(url);
```

```
rd.forward(request, response);
```

Controller (Servlet) → SESSION → View (JSP)

- View (JSP)
 - Java Bean

```
<jsp:useBean id="key"  
    type="ArrayList<POJO>" scope="session"/>
```

. . .

- Scriptlet

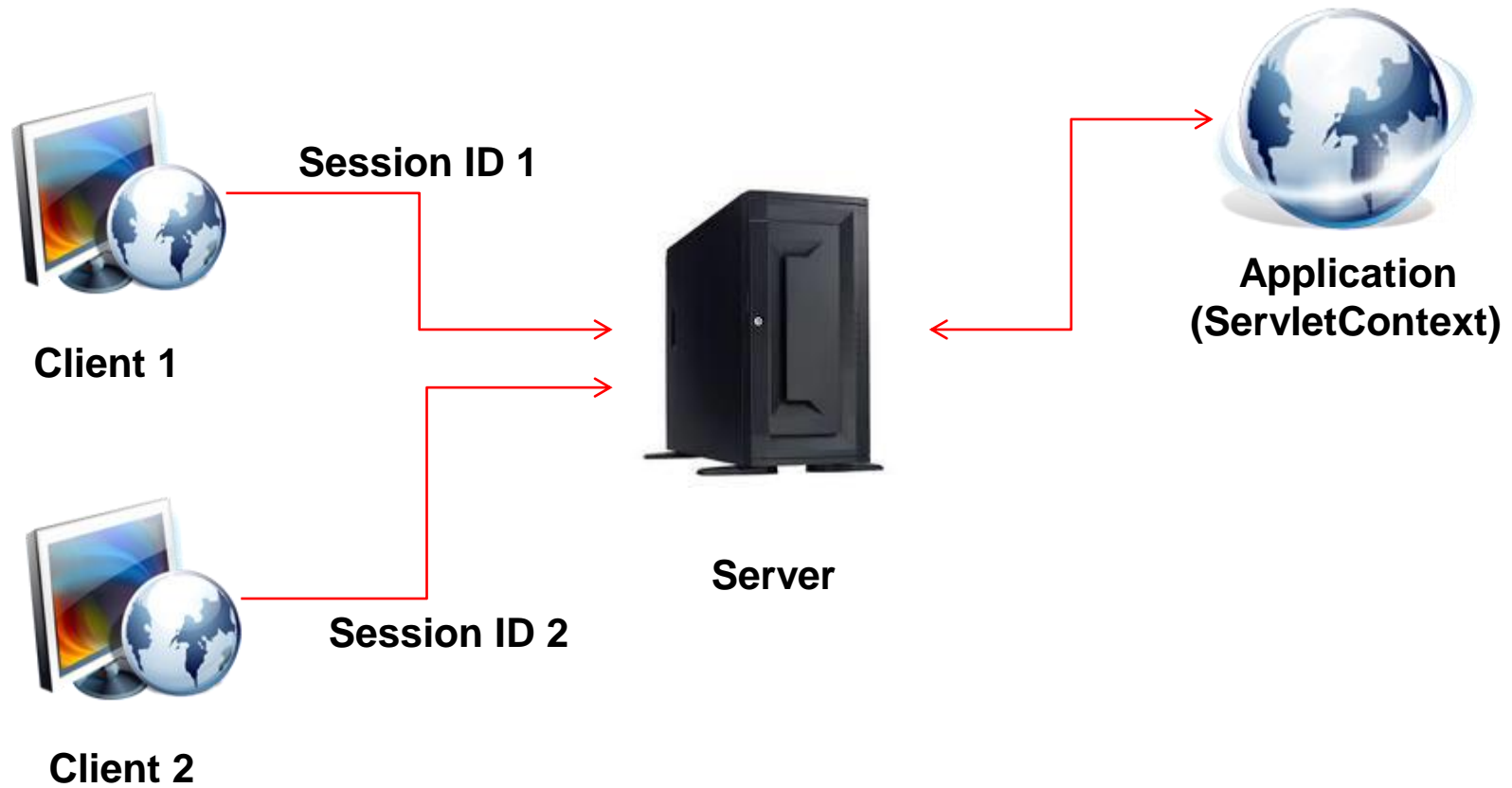
```
<% ArrayList<POJO> ds=  
(ArrayList<POJO>)session.getAttribute("Key");  
%>
```

Các ví dụ khác

- Quản lý giỏ hàng

Controller (Servlet) → **APPLICATION** → View (JSP)

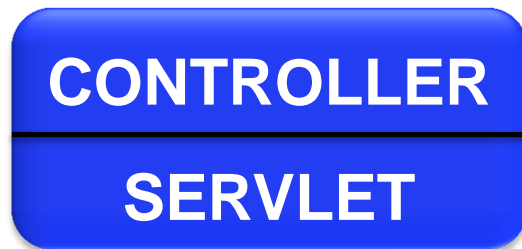
- Session



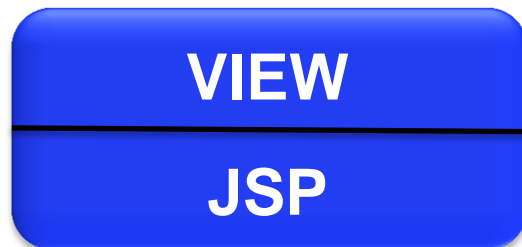
Controller (Servlet) → APPLICATION → View (JSP)

- Forward
 - Người dùng không nhìn thấy URL thay đổi
 - Người dùng không thể Bookmark
 - Quá trình chuyển dữ liệu Servlet → JSP
- SendRedirect
 - Người dùng nhìn thấy URL thay đổi
 - Người dùng có thể Bookmark trang JSP
 - Quá trình chuyển dữ liệu JSP → Servlet → JSP
 - Bên View cần kiểm tra dữ liệu đã tồn tại chưa

Controller (Servlet) → APPLICATION → View (JSP)



Forward



```
synchronized(this) {  
    Object value1 = ...  
    Object value2 = ...  
    ServletContext application  
        = this.getServletContext();  
    application.setAttribute ("Key1", value)  
    application.setAttribute ("Key2", value)  
    ...  
}
```

```
Object v1 = application.getAttribute ("Key1")  
Object v2 = application.getAttribute ("Key2")  
...
```

Controller (Servlet) → APPLICATION → View (JSP)

- Controller (Servlet)

```
synchronized(this) {  
    POJO pojo = . . .  
    ServletContext application  
                                   = this.getServletContext();  
    application.setAttribute("key", pojo);  
    String url="url view";  
    RequestDispatcher rd =  
        request.getRequestDispatcher(url);  
    rd.forward(request, response);  
}
```


Controller (Servlet) → APPLICATION → View (JSP)

- View (JSP)
 - Java Bean

```
<jsp:useBean id="key"  
              type="POJO" scope="application"/>
```

. . .

- Scriptlet

```
<% POJO pojo=  
    (POJO)application.getAttribute("Key");  
%>
```

Controller (Servlet) → APPLICATION → View (JSP)

- Controller (Servlet)

```
synchronized(this) {  
    ArrayList<POJO> ds = . . .  
    ServletContext application  
                                = this.getServletContext();  
    application.setAttribute("key", ds);  
    String url="url view";  
    RequestDispatcher rd =  
        request.getRequestDispatcher(url);  
    rd.forward(request, response);  
}
```

Controller (Servlet) → APPLICATION → View (JSP)

- View (JSP)
 - Java Bean

```
<jsp:useBean id="key" type="ArrayList<POJO>"  
             scope="application"/>
```

- Scriptlet

```
<% ArrayList<POJO> ds=  
    (ArrayList<POJO>)application.getAttribute("Key");  
%>
```

Controller (Servlet) gửi dữ liệu cho View (JSP)

- Request
- Session
- Application (ServletContext)

MVC 2





HỎI VÀ ĐÁP

Reference

- [1] Marty Hall; Larry Brown; Yaakov Chaikin. (2007). Core Servlets and Javaserver Pages: Advanced Technologies, Vol. 2, 2nd Edition. Prentice Hall.
- [2] Joel Murach; Michael Urban. (2014). Murach's Java Servlets and JSP, 3rd Edition. Mike Murach & Associate.