



**SERVLET**

ThS. Nguyễn Nghiệm  
0913.745.789 - NghiemN@fpt.edu.vn



# NỘI DUNG

---

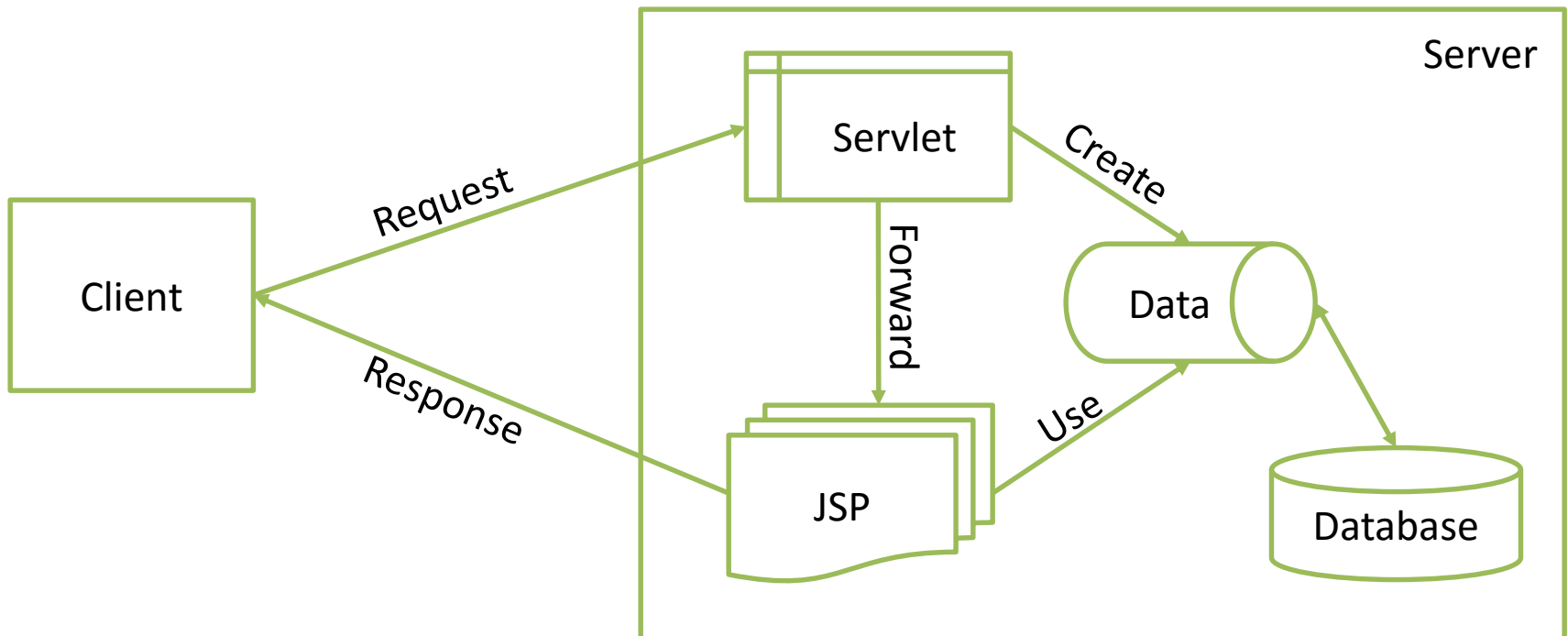
- Mô hình MVC
- Servlet
- Parameter
- Cookie
- Chia sẻ dữ liệu
- Bộ lọc – Filter
- Bộ nghe – Listener
- Đóng gói và triển khai ứng dụng





# Mô HÌNH MVC

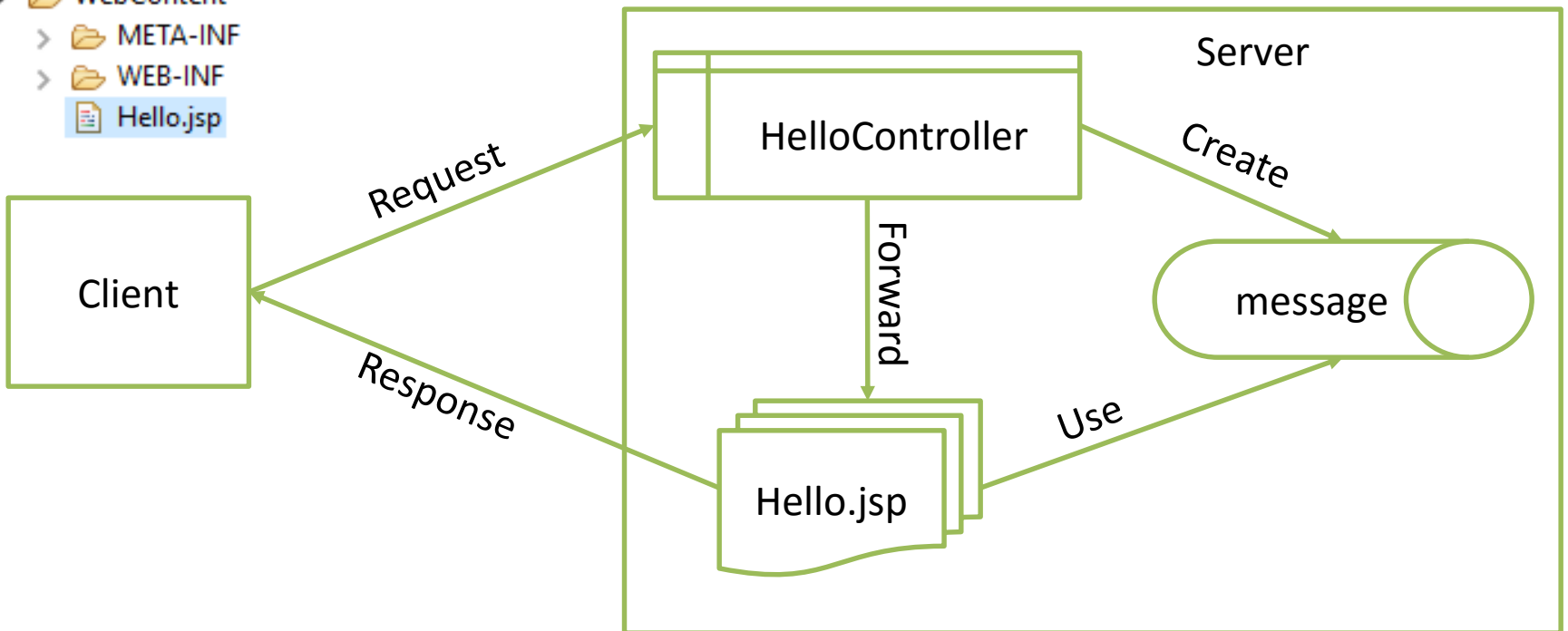
- Controller: Servlet
- View: JSP
- Model: Dữ liệu và chia sẻ





# HELLO MVC

- Slides
  - > JAX-WS Web Services
  - ▼ Java Resources
    - ▼ src
      - ▼ ngghiemn.servlet
        - > HelloController.java
      - > Libraries
    - > JavaScript Resources
    - > build
    - ▼ WebContent
      - > META-INF
      - > WEB-INF
        - Hello.jsp





# HELLO MVC



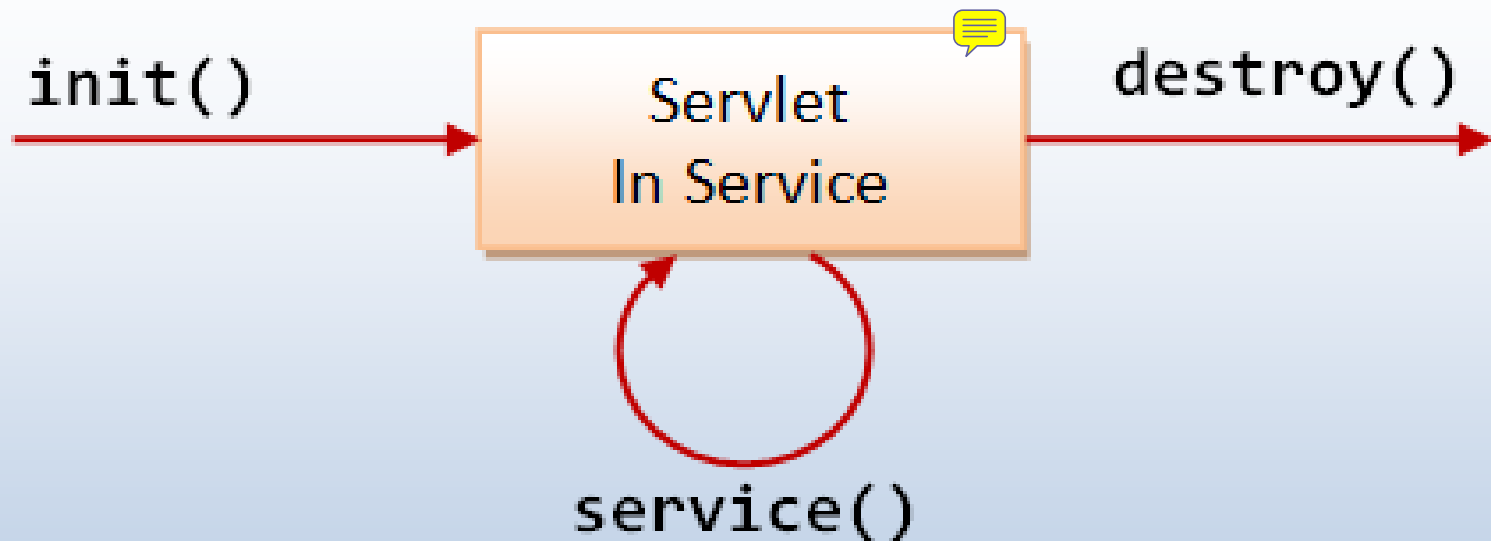
```
@WebServlet("/hello.do")
public class HelloController extends HttpServlet{
    @Override
    protected void service(HttpServletRequest req, HttpServletResponse resp)
        throws ServletException, IOException {
        req.setAttribute("message", "Hello Servlet & JSP");
        req.getRequestDispatcher("Hello.jsp").forward(req, resp);
    }
}
```

```
<%@ page pageEncoding="utf-8"%>
<!DOCTYPE html>
<html>
<head>
    <meta charset="utf-8">
    <title>Insert title here</title>
</head>
<body>
    ${message}
</body>
</html>
```



# SERVLET LIFECYCLE

## Servlet Container



**HTTP:**

`doGet()`, `doPost()`, `doHead()`,  
`doOptions()`, `doTrace()`, etc.



# CẤU TRÚC

---

```
//imports libraries
public class HitCounterServlet extends HttpServlet {
    // chứa số lần truy xuất
    private int counter = 0;
    protected void doGet(HttpServletRequest request,
        HttpServletResponse response) throws ServletException, IOException {
        // tăng số đếm lên 1
    }
    protected void doPost(HttpServletRequest request,
        HttpServletResponse response) throws ServletException, IOException {
        // tăng số đếm lên 1
    }
    public void init(ServletConfig config) throws ServletException {
        // đọc số đếm từ file HitCounter.txt
    }
    public void destroy() {
        // ghi số đếm vào file HitCounter.txt
    }
}
```



# KHAI BÁO SERVLET

---

## ● web.xml

```
<servlet-mapping>  
  <servlet-name>HelloServlet</servlet-name>  
  <url-pattern>/hello.do</url-pattern>  
</servlet-mapping>
```

```
<welcome-file-list>  
  <welcome-file>index.jsp</welcome-file>  
</welcome-file-list>
```

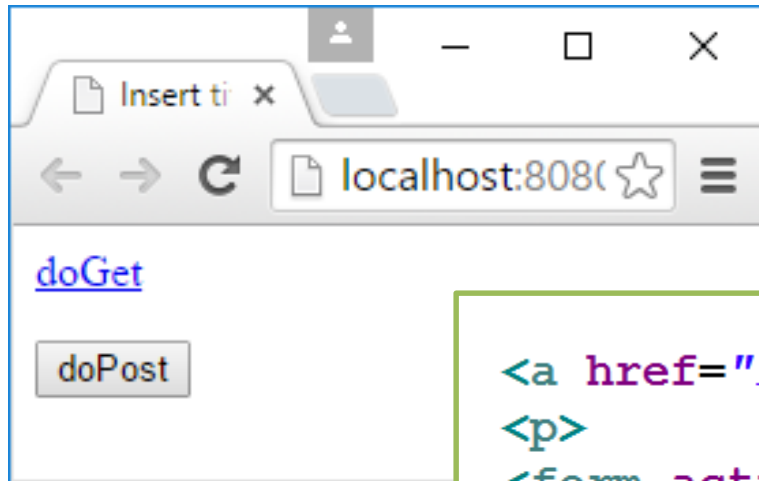
## ● Annotation (với Servlet 3.0+)

```
@WebServlet("/hello.do")  
public class HelloController extends HttpServlet{
```





# TRUY CẬP SERVLET



```
<a href="hello.do">doGet</a>
<p>
<form action="hello.do" method="post">
  <button>doPost</button>
</form>
```



# ĐỀ MÔ - ACCESSING COUNTER

---

```
@WebServlet("/access.do")
public class AccessCounter extends HttpServlet{
    int counter = 0;

    @Override
    public void init() throws ServletException {
        // đọc số truy cập từ file
    }
    @Override
    public void destroy() {
        // lưu số truy cập vào file
    }
    @Override
    protected void service(HttpServletRequest req, HttpServletResponse resp)
        throws ServletException, IOException {
        // tăng số truy cập
    }
}
```



# PARAMETER

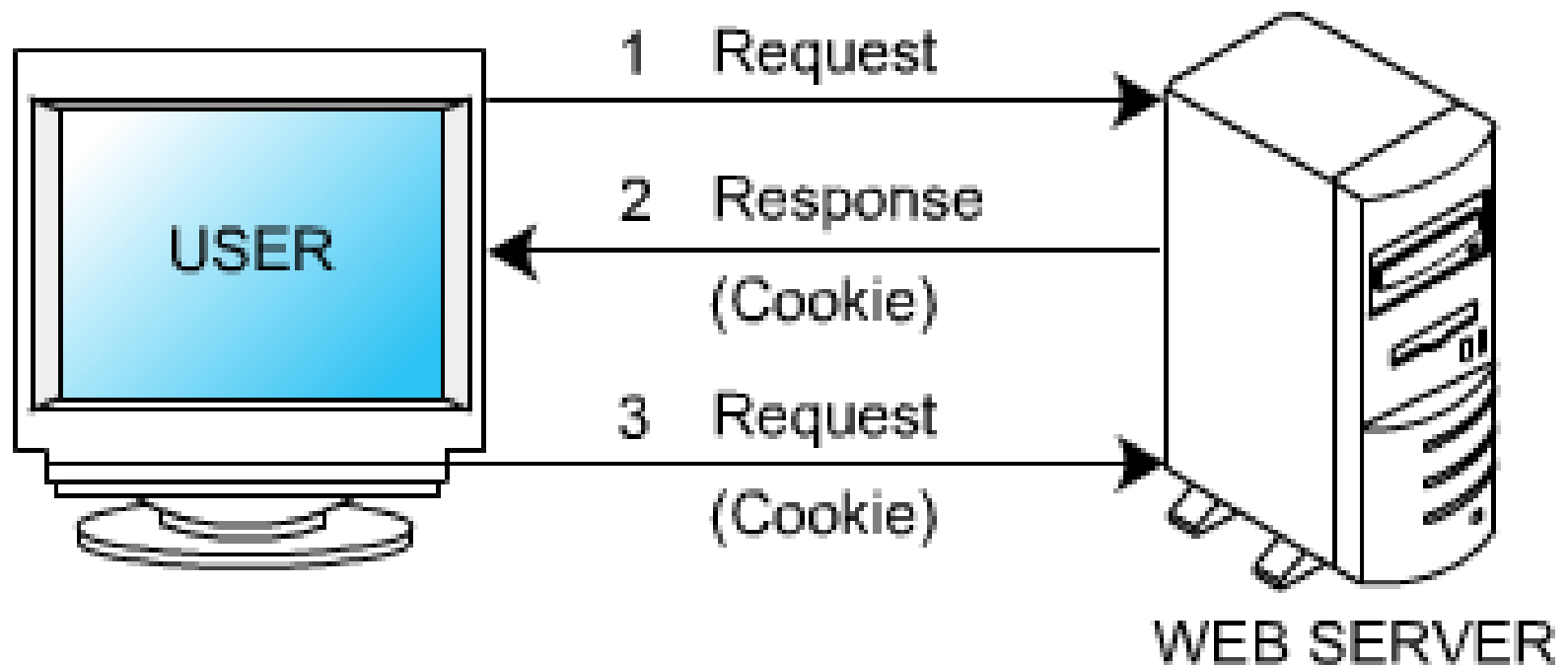
---

- Lấy giá trị tham số
  - ✱ `request.getParameter(name)`
- Lấy giá trị nhiều tham số cùng tên
  - ✱ `request.getParameterValues(name)`
- Lấy danh sách tất cả các tên tham số
  - ✱ `request.getParameterNames()`
- Ví dụ
  - ✱ `String[] hobbies = request.getParameterValues("hobby")`



# COOKIE

- Cookie là mẫu tin văn bản nhỏ được lưu trên máy client và chỉ tồn tại trong một khoảng thời gian nhất định. Nó được gửi lên server theo request.





# LÀM VIỆC VỚI COOKIE

```
String name = "MyCookie";  
String value = "My Cookie Value";  
int maxAge = 60*60*24*365; // 1 năm  
Cookie newCookie = new Cookie(name, value);  
newCookie.setMaxAge(maxAge);  
response.addCookie(newCookie);
```



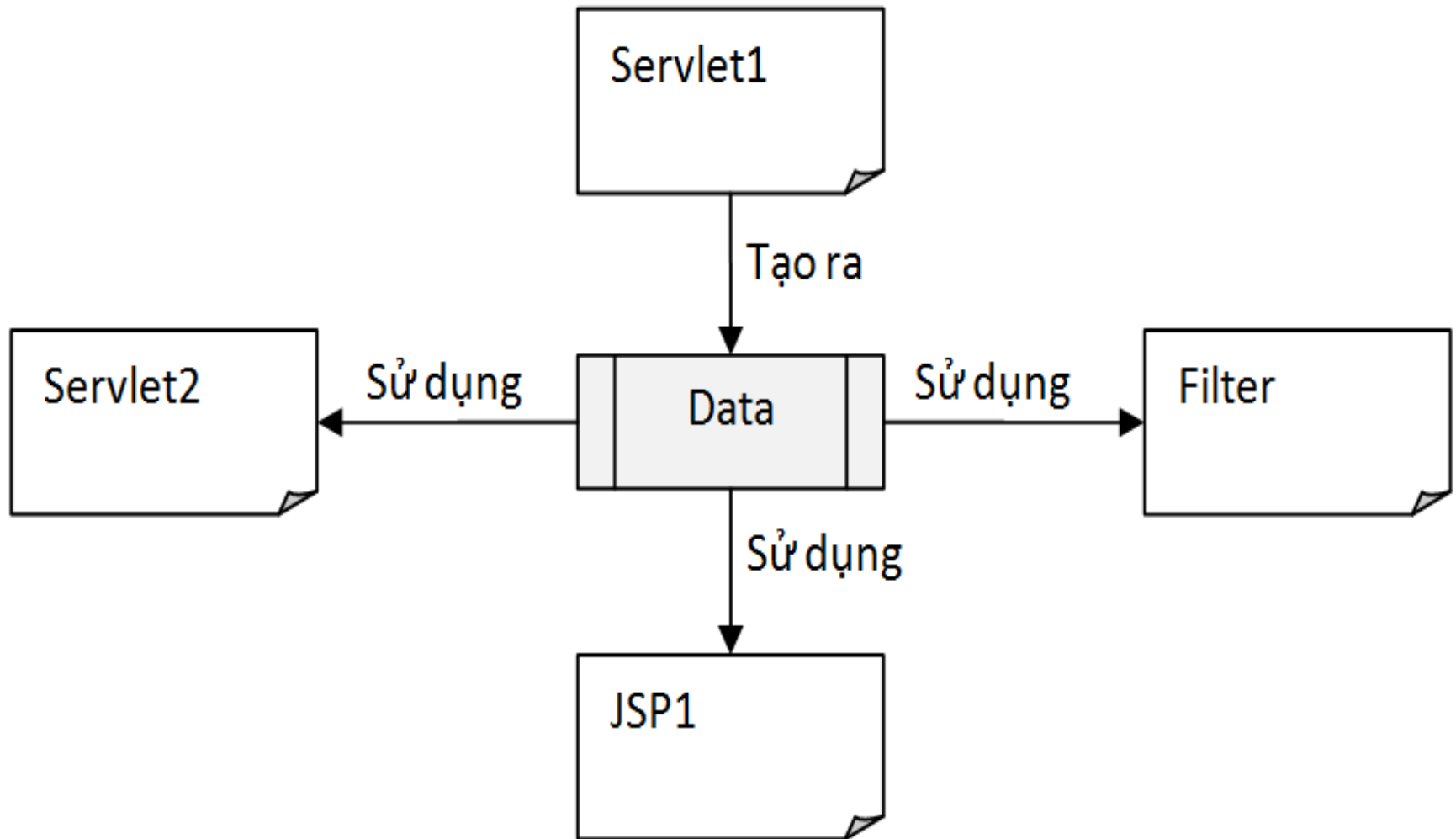
Tạo, thiết lập thời gian tồn tại và gửi cookie về client để lưu lại

```
Cookie[] cookies = request.getCookies();  
for(int i=0; cookies != null && i<cookies.length; i++)  
{  
    Cookie cookie = cookies[i];  
    String name = cookie.getName();  
    String value = cookie.getValue();  
}
```

Đọc các cookie được gửi lên server theo request



# CHIA SẺ DỮ LIỆU





# PHẠM VI CHIA SẺ

## ● 3 scope

### ✱ Application (ServletContext)

- Được sử dụng để chia sẻ dữ liệu giữa các thành phần web trên phạm vi toàn ứng dụng

### ✱ Session (HttpSession)

- Chia sẻ dữ liệu giữa các thành phần web trong phạm vi mỗi phiên làm việc

### ✱ Request (HttpServletRequest)

- Chia sẻ dữ liệu giữa các thành phần web hoạt động trên cùng một yêu cầu.

## ● Tham chiếu đến scope

✱ HttpSession session= **request.getSession()**

✱ ServletContext application= **this.getServletContext()**



# <SCOPE> API

---

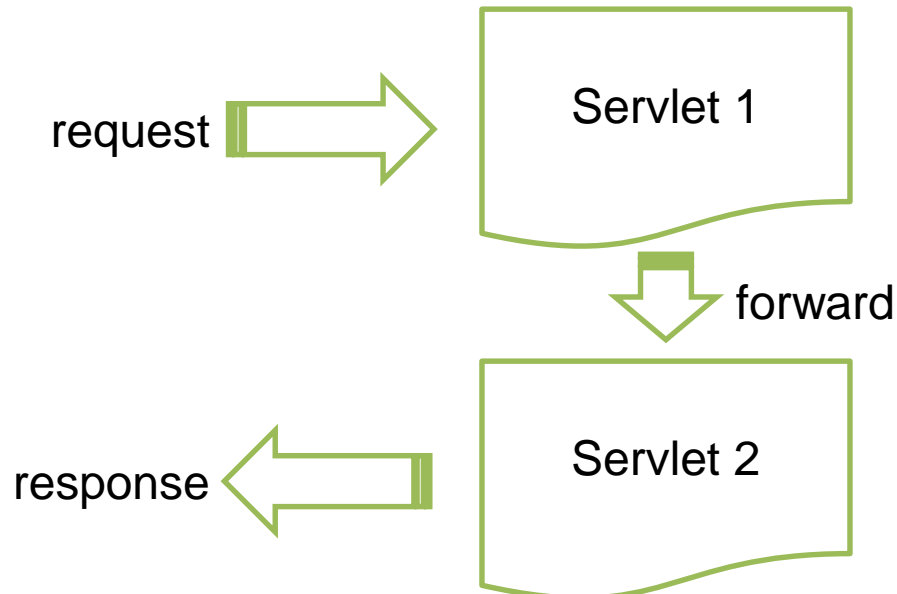
- `<scope>.setAttribute(String name, Object value)`
  - ✱ Tạo mới hoặc thay thế thuộc tính
- `Object <scope>.getAttribute(String name)`
  - ✱ Lấy giá trị thuộc tính
- `<scope>.removeAttribute(String name)`
  - ✱ Xóa thuộc tính
- `Enumeration<String>`  
`<scope>.getAttributeNames()`
  - ✱ Lấy danh sách tất cả tên của thuộc tính





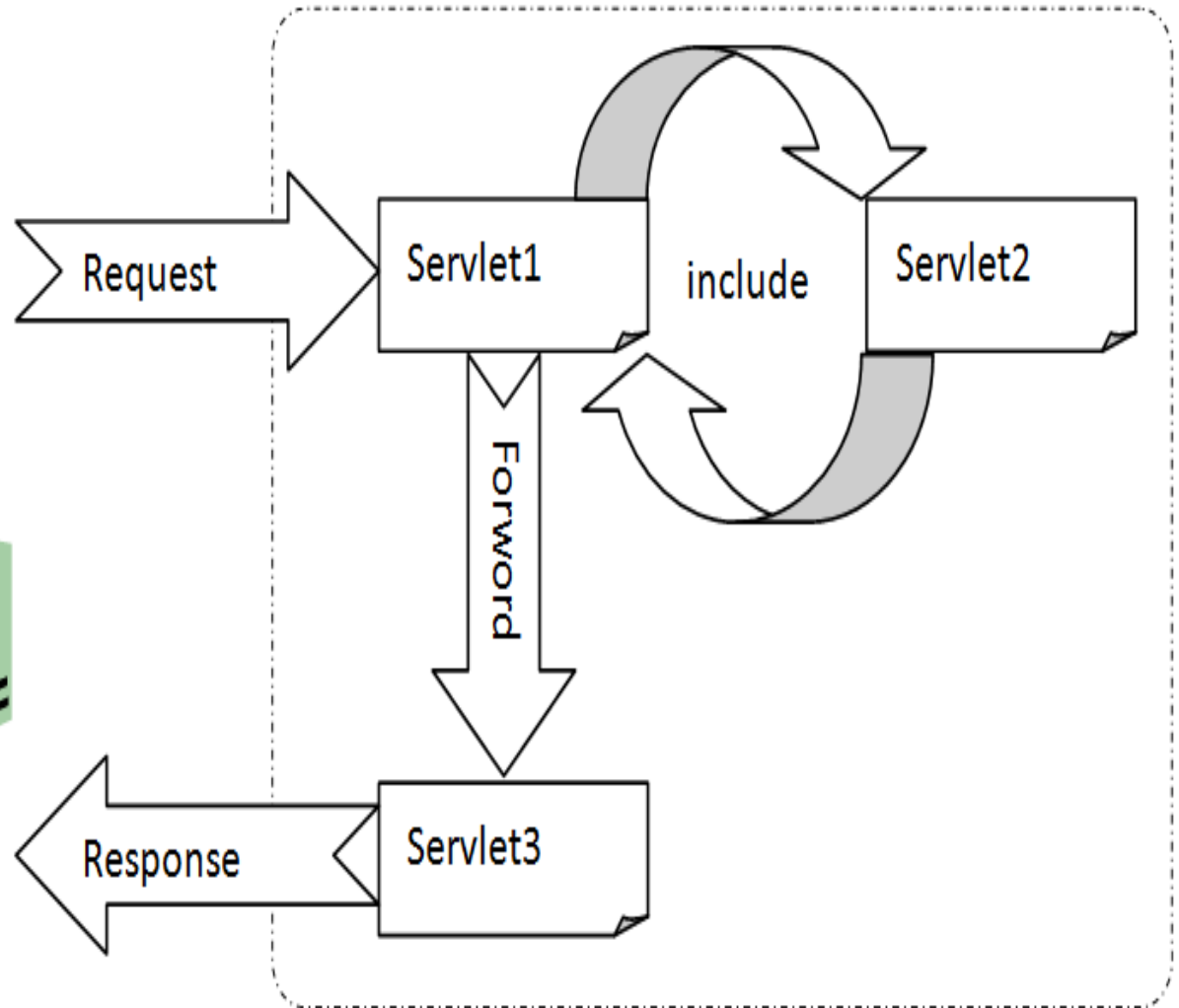
# REQUEST

- `req.getRequestDispatcher("hello.do").forward(req, resp)`
- `req.getRequestDispatcher("hello.do").include(req, resp)`





# REQUEST SCOPE





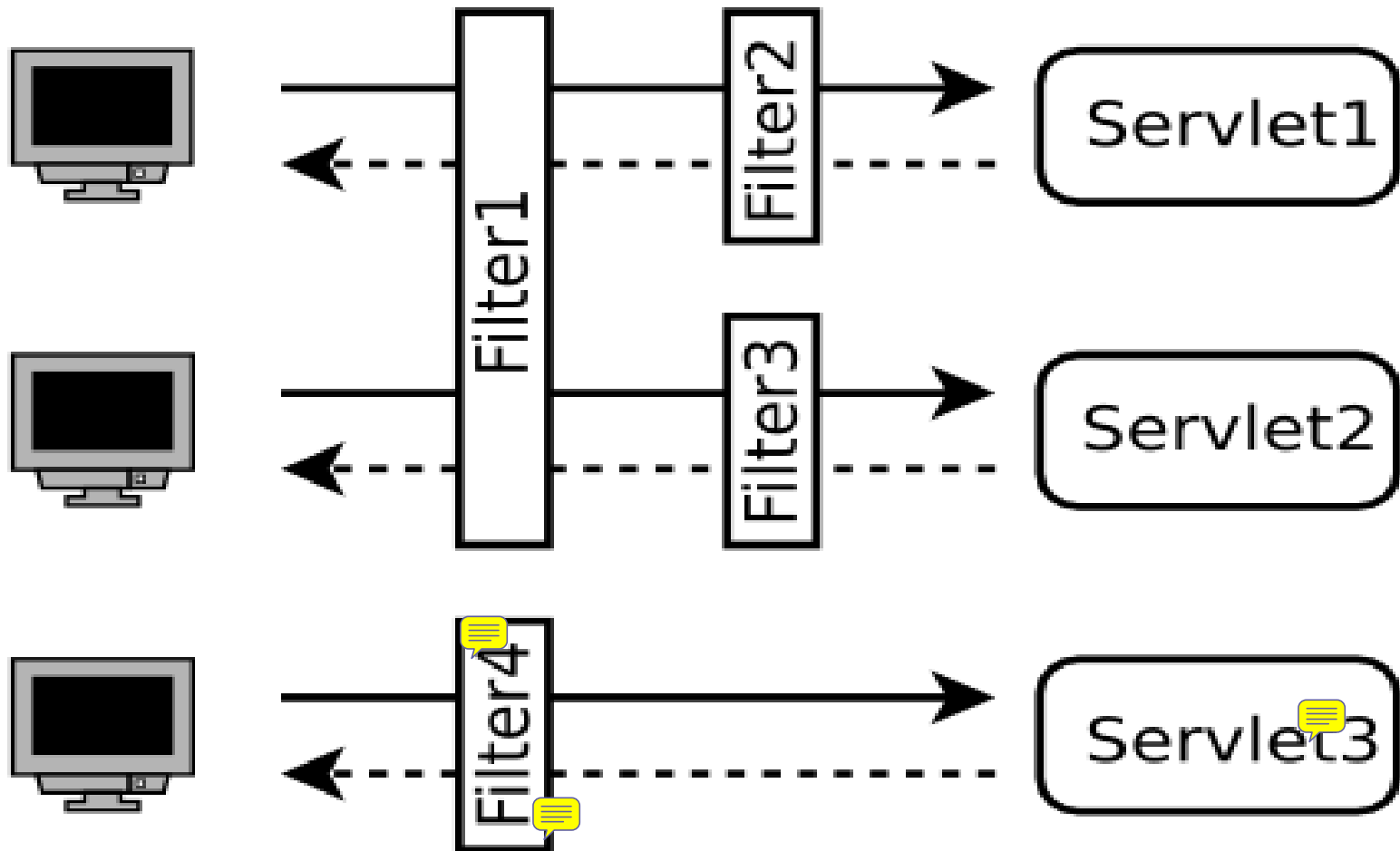
# APPLICATION & SESSION SCOPE

```
ServletContext application = getServletContext();
if(application.getAttribute("counter") == null)
{
    application.setAttribute("counter", 2011);
}
else
{
    int counter = (Integer)application.getAttribute("counter");
    application.setAttribute("counter", counter + 1);
}
```

```
HttpSession session = request.getSession();
if(session.getAttribute("user") == null)
{
    response.sendRedirect("DangNhapServlet.do");
}
```



# FILTER





```
public class MyFilter implements Filter{

    @Override
    public void destroy() {
        System.out.println(" >> MyFilter.destroy()");
    }

    @Override
    public void doFilter(ServletRequest req, ServletResponse resp,
        FilterChain chain) throws IOException, ServletException {
        System.out.println(" >> MyFilter.doFilter()-before");
        chain.doFilter(req, resp);
        System.out.println(" >> MyFilter.doFilter()-after");
    }

    @Override
    public void init(FilterConfig config) throws ServletException {
        System.out.println(" >> MyFilter.init()");
    }
}
```



# KHAI BÁO FILTER

## ● Web.xml

```
<filter>
  <filter-name>MyFilter</filter-name>
  <filter-class>nghiemn.filter.MyFilter</filter-class>
</filter>

<filter-mapping>
  <filter-name>MyFilter</filter-name>
  <url-pattern>/*</url-pattern>
  <dispatcher>REQUEST</dispatcher>
</filter-mapping>
```

## ● Annotation

```
@WebFilter(urlPatterns={"/*"},
    dispatcherTypes={DispatcherType.REQUEST})
public class MyFilter implements Filter{
```



# ĐỀ MÔ - UNICODE

```
@WebFilter(urlPatterns={"/*"},
    dispatcherTypes={DispatcherType.REQUEST})
public class UTF8Filter implements Filter{
    @Override
    public void doFilter(ServletRequest request, ServletResponse response,
        FilterChain chain) throws IOException, ServletException {

        request.setCharacterEncoding("utf-8");
        response.setCharacterEncoding("utf-8");

        chain.doFilter(request, response);
    }

    @Override
    public void init(FilterConfig config) throws ServletException {}

    @Override
    public void destroy() {}
}
```



# ĐỀ MÔ – HITCOUNTER

---

```
@WebFilter(urlPatterns={"/*"},
    dispatcherTypes={DispatcherType.REQUEST})
public class HitCounterFilter implements Filter{
    String path = "hits.properties";
    Properties hits = new Properties();

    @Override
    public void init(FilterConfig config) throws ServletException {
        // Tải file lưu số đếm hits.properties vào Properties
    }
    @Override
    public void destroy() {
        // Lưu Properties vào file hits.properties
    }
    @Override
    public void doFilter(ServletRequest request, ServletResponse response,
        FilterChain chain) throws IOException, ServletException {
        // code
    }
}
```





# ĐỀ MÔ – HITCOUNTER

```
try {  
    path = config.getServletContext().getRealPath(path);  
    hits.load(new FileReader(path));  
}  
catch (Exception e) {  
    e.printStackTrace();  
}  
config.getServletContext().setAttribute("hits", hits);
```

init()

```
try {  
    hits.store(new FileWriter(path), "");  
}  
catch (Exception e) {  
    e.printStackTrace();  
}
```

destroy()

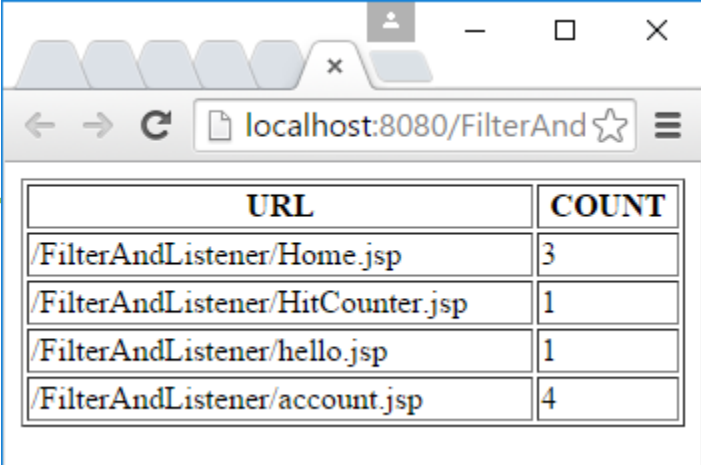
```
HttpServletRequest req = (HttpServletRequest) request;  
String uri = req.getRequestURI();  
  
// Tăng số đếm của trang đang được truy cập  
Integer count = Integer.valueOf(hits.getProperty(uri, "0")) + 1;  
hits.setProperty(uri, String.valueOf(count));  
  
chain.doFilter(request, response);
```

doFilter()



# HIỂN THỊ HITCOUNTER

```
<table border="1" style="width:100%">
<tr>
    <th>URL</th>
    <th>COUNT</th>
</tr>
<c:forEach var="e" items="${hits}">
<tr>
    <td>${e.key}</td>
    <td>${e.value}</td>
</tr>
</c:forEach>
</table>
```



A screenshot of a web browser window displaying the output of the JSP code. The browser's address bar shows 'localhost:8080/FilterAnd'. The page content is a table with two columns: 'URL' and 'COUNT'. The table lists four entries: '/FilterAndListener/Home.jsp' with a count of 3, '/FilterAndListener/HitCounter.jsp' with a count of 1, '/FilterAndListener/hello.jsp' with a count of 1, and '/FilterAndListener/account.jsp' with a count of 4.

URL	COUNT
/FilterAndListener/Home.jsp	3
/FilterAndListener/HitCounter.jsp	1
/FilterAndListener/hello.jsp	1
/FilterAndListener/account.jsp	4



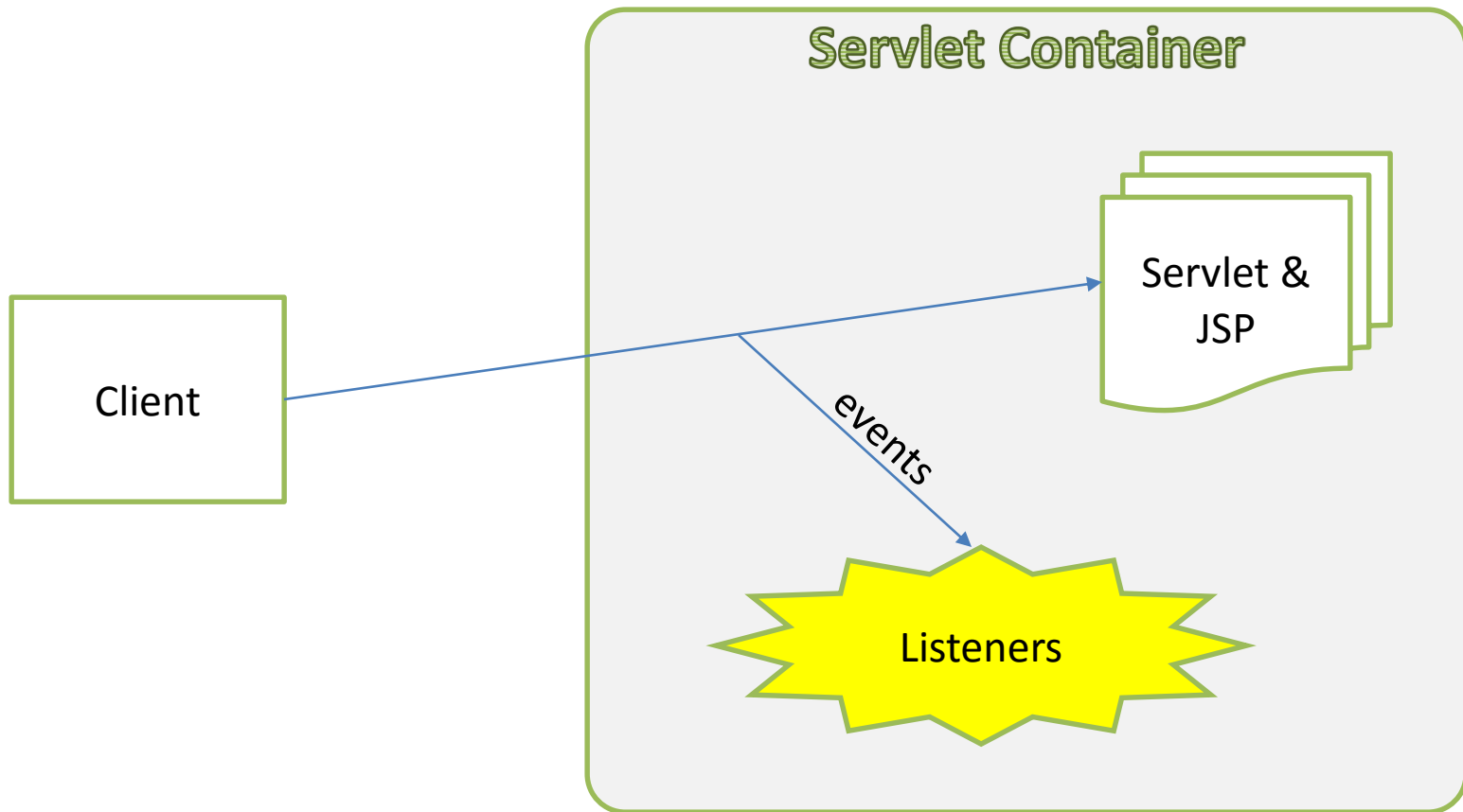
# THỨ TỰ THỰC HIỆN

---

```
<web-app>
  <!-- bộ lọc SecurityFilter chạy đầu tiên -->
  <filter>
    <filter-name>SecurityFilter</filter-name>
    <filter-class>nnghiem.nhatnghe.filter.SecurityFilter</filter-class>
  </filter>
  <filter-mapping>
    <filter-name>SecurityFilter</filter-name>
    <url-pattern>*</url-pattern>
    <dispatcher>REQUEST</dispatcher>
  </filter-mapping>
  <!-- bộ lọc LayoutFilter chạy sau bộ lọc SecurityFilter -->
  <filter>
    <filter-name>LayoutFilter</filter-name>
    <filter-class>nnghiem.nhatnghe.filter.LayoutFilter</filter-class>
  </filter>
  <filter-mapping>
    <filter-name>MyFilter</filter-name>
    <url-pattern>*.jsp</url-pattern>
    <dispatcher>REQUEST</dispatcher>
    <dispatcher>FORWARD</dispatcher>
  </filter-mapping>
</web-app>
```



# LISTENER





# KIỂM SOÁT SESSION

@WebListener

```
public class AppListener implements HttpSessionListener {

    /**
     * Chay ngay sau khi một phiên bắt đầu
     */
    @Override
    public void sessionCreated(HttpSessionEvent e) {
        HttpSession session = e.getSession();
        ServletContext application = session.getServletContext();
    }

    /**
     * Chay ngay trước khi một phiên bị session timeout
     */
    @Override
    public void sessionDestroyed(HttpSessionEvent e) {
        HttpSession session = e.getSession();
        ServletContext application = session.getServletContext();
    }
}
```



# KIỂM SOÁT APPLICATION

@WebListener

```
public class AppListener implements ServletContextListener {

    /**
     * Chay ngay trước khi ứng dụng shutdown
     */
    @Override
    public void contextDestroyed(ServletContextEvent e) {
        ServletContext application = e.getServletContext();
    }

    /**
     * Chay ngay sau khi ứng dụng khởi động
     */
    @Override
    public void contextInitialized(ServletContextEvent e) {
        ServletContext application = e.getServletContext();
    }
}
```



# ĐỀ MÔ – VISITORS COUNTER

---

- Tạo 1 listener để nghe 4 sự kiện: Ứng dụng bắt đầu, ứng dụng kết thúc, session bắt đầu và session kết thúc và viết mã cho các sự kiện:
- Ứng dụng bắt đầu:
  - ✱ Đọc số khách truy cập từ file và lưu vào application
- Ứng dụng kết thúc:
  - ✱ Lưu số khách truy cập từ application vào file
- Session bắt đầu:
  - ✱ Tăng số khách truy cập trong application lên 1
- Trang giao diện:
  - ✱ Hiển thị số khách truy cập từ application lên giao diện



# VISITORS COUNTER

---

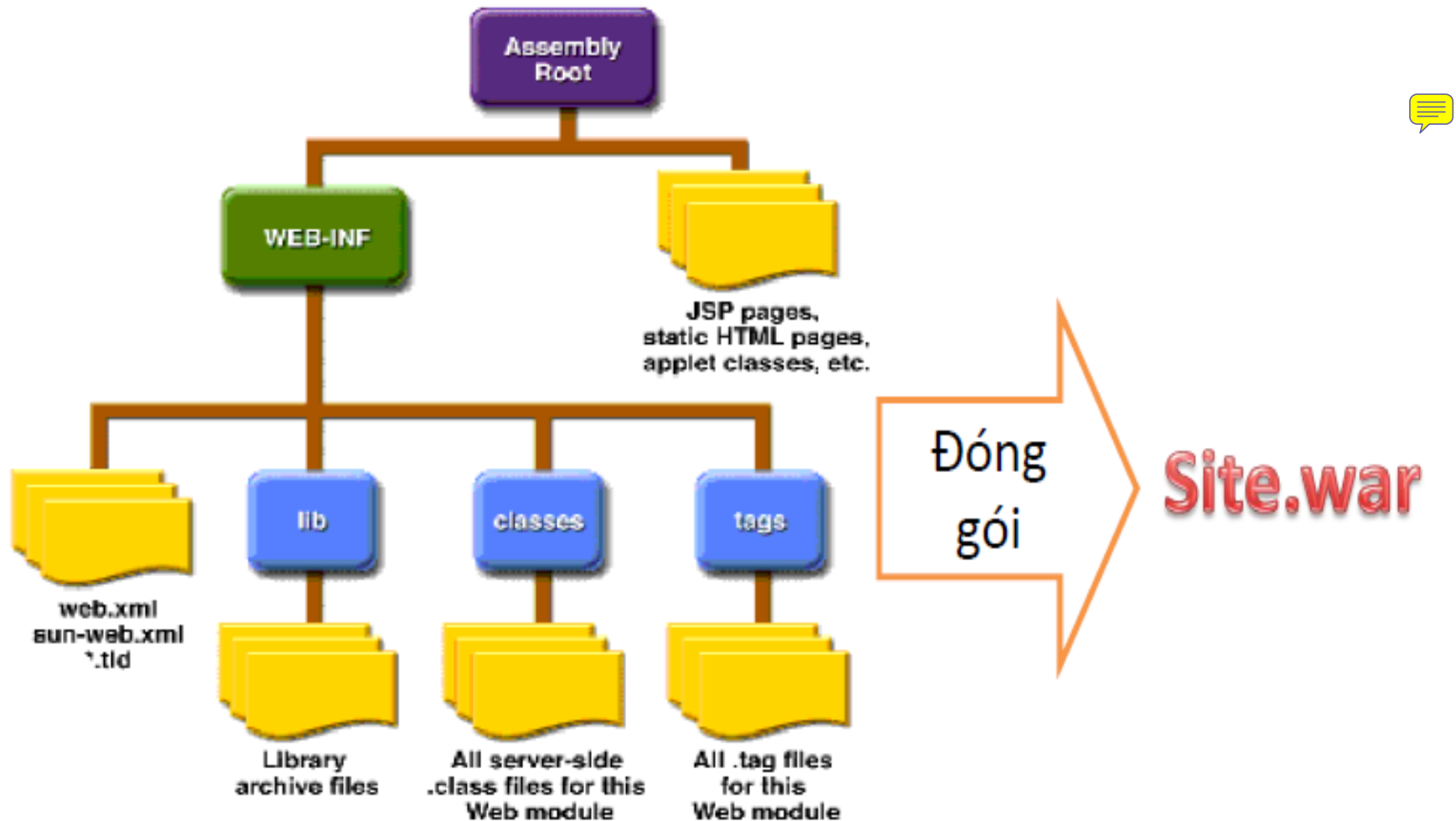
```
@WebListener
public class AppListener
    implements HttpSessionListener, ServletContextListener{

    @Override
    public void contextInitialized(ServletContextEvent e) {
        // Tải số khách truy cập từ file
    }
    @Override
    public void contextDestroyed(ServletContextEvent e) {
        // Lưu số khách truy cập vào file
    }
    @Override
    public void sessionCreated(HttpSessionEvent e) {
        // Tăng số khách truy cập
    }
    @Override
    public void sessionDestroyed(HttpSessionEvent e) {
    }
}
```





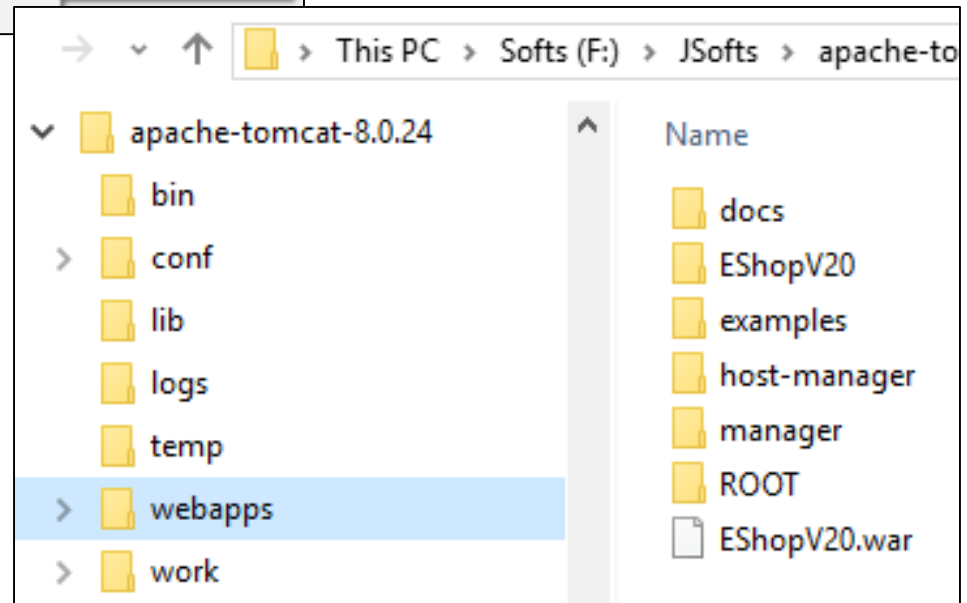
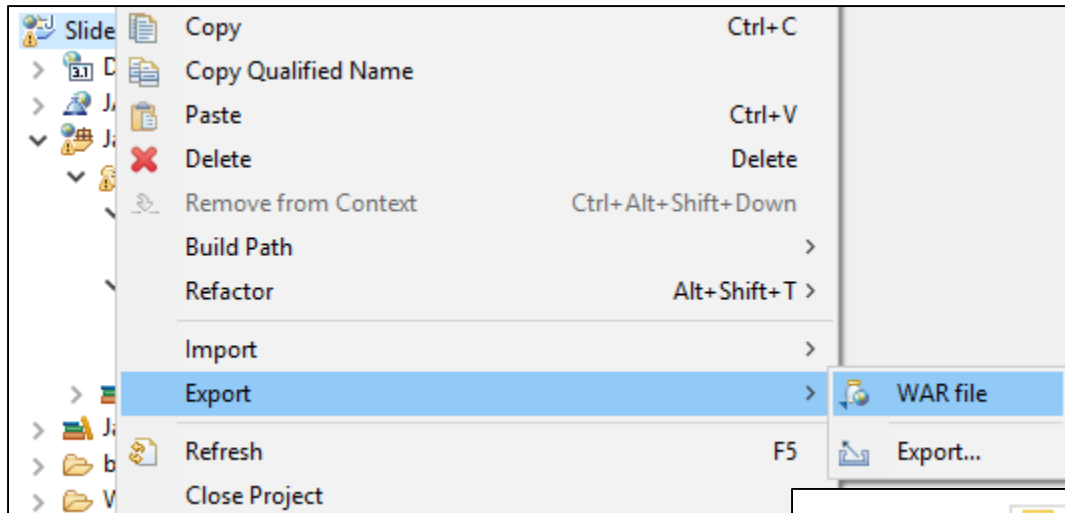
# ĐÓNG GÓI ỨNG DỤNG WEB



- Triển khai: Chép thư mục Site hoặc tập tin Site.war vào thư mục webapps của Tomcat



# TRIỂN KHAI ỨNG DỤNG WEB





# TÓM TẮT

---

- Mô hình MVC
- Servlet
- Parameter
- Cookie
- Chia sẻ dữ liệu
- Bộ lọc – Filter
- Bộ nghe – Listener
- Đóng gói và triển khai ứng dụng