

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

# LISTENER

**Nguyễn Hoàng Anh**  
**[nhanh@fit.hcmus.edu.vn](mailto:nhanh@fit.hcmus.edu.vn)**

# Nội dung

- **Lắng nghe các event ở application scope**
- **Lắng nghe các event ở session scope**
- **Lắng nghe các event ở request scope**

# Listener

- Một **listener** là một **class** được dùng để **lắng nghe** các **event** khác nhau trong suốt chu kỳ sống của một **web application** và **cung cấp** các **phương thức** để **trả lời** các **event** tương ứng xảy ra.
- Để **tạo** một **class** cho một **listener** cần **implement** lại **một** trong các **listener interface** trong các package là **javax.servlet** và **javax.servlet.http**
- Một class implement từ một listener interface phải override lại tất cả các phương thức của interface đó.

# LẮNG NGHE CÁC EVENT Ở APPLICATION SCOPE

# Lắng nghe các event ở application scope

- **Lắng nghe 2 event** liên quan đến **web application** được **start** hoặc **stop**:
  - `javax.servlet.ServletContextListener`
- **Lắng nghe các event** liên quan đến sự **thay đổi** các **attribute** trong **application scope**:
  - `javax.servlet.ServletContextAttributeListener`

# Lắng nghe các event ở application scope

- **Lắng nghe 2 event** liên quan đến **web application** được **start** hoặc **stop**:
  - `javax.servlet.ServletContextListener`

```
1 public interface ServletContextListener {  
2  
3     public void contextInitialized(ServletContextEvent event);  
4  
5     public void contextDestroyed(ServletContextEvent event);  
6 }
```

# ServletContextListener

- **ServletContextListener** lắng nghe 2 event xảy ra ở **application scope**:
  - **Web application** được **start**.
  - **Web application** được **stop**.
- **ServletContextListener** cung cấp 2 phương thức tương ứng:
  - (1) **contextInitialized**: được tự động gọi khi **web application** được **start**.
  - (2) **contextDestroyed**: được tự động gọi khi **web application** được **stop**.

# ServletContextListener

- **Bước 1:** Cài đặt listener.
- **Bước 2:** Đăng ký listener.
- **Bước 3:** Sử dụng các attribute.



# Bước 1: Cài đặt listener

```
1 public class MyAppContextListener implements ServletContextListener {
2
3     public void contextInitialized(ServletContextEvent event) {
4         ServletContext ctx = event.getServletContext();
5         Object value1 = ...;
6         Object value2= ...;
7         ctx.setAttribute("key1", value1);
8         ctx.setAttribute("key2", value2);
9         System.out.println("key1, key2 initialized for Application.");
10    }
11    public void contextDestroyed(ServletContextEvent event) {
12        ServletContext sc = event.getServletContext();
13        Object value1 =sc.getAttribute("key1"); //close
14        Object value2= sc.getAttribute("key2"); //close
15        System.out.println("key1, key2 closed for Application.");
16    }
17 }
```

## Bước 2: Đăng ký listener

- (1) Khai báo bằng **Annotation**

**@WebListener**

```
public class MyAppContextListener implements ServletContextListener
```

- (2) Khai báo trong **web.xml**

```
<web-app ...>
  <display-name>application-context-listener-sample</display-
name>
  <listener>
    <listener-class>
      org.nhanh.listener.MyAppContextListener
    </listener-class>
  </listener>
</web-app>
```

## Bước 3: Sử dụng các attribute

- **Controller** (Servlet)

```
ServletContext app = request.getServletContext();
```

```
Object value1 = app.getAttribute("key1");
```

```
Object value2 = app.getAttribute("key2");
```

- **View** (JSP)

```
${applicationScope.key1}
```

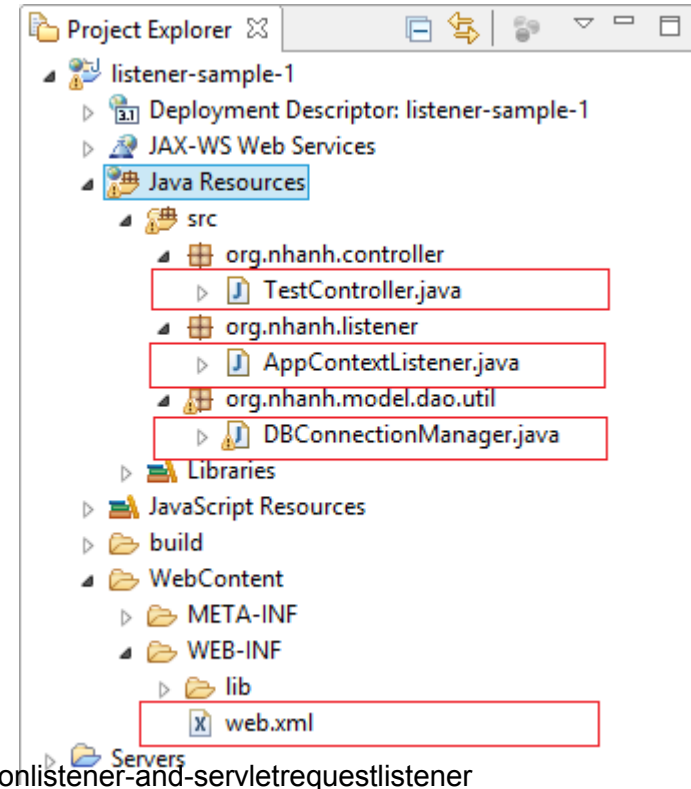
```
${applicationScope.key2}
```

# Demo



# ServletContextListener

- Ví dụ: Quản lý một database connection cho web application ở application scope.
  - Bước 0: Xây dựng DBConnectionManager
  - Bước 1: Cài đặt listener.
  - Bước 2: Đăng ký listener.
  - Bước 3: Sử dụng các attribute.



[1] <http://www.journaldev.com/1945/servlet-listener-example-servletcontextlistener-httpsessionlistener-and-servletrequestlistener>

# Bước 0: Xây dựng DBConnectionManager

```
*DBConnectionManager.java
1 package org.nhanh.model.dao.util;
2 import java.sql.Connection;
7 public class DBConnectionManager {
8     private String dbUser;
9     private String dbPassword;
10    private String dbURL;
11    private String dbCharacterEncoding;
12    private String dbDriver;
13    private Connection connection;
14    public DBConnectionManager(String dbDriver, String dbUser, String dbPassword, String dbURL,
15        String dbCharacterEncoding) throws ClassNotFoundException, SQLException {
16        this.dbDriver=dbDriver;
17        this.dbUser = dbUser;
18        this.dbPassword = dbPassword;
19        this.dbURL = dbURL;
20        this.dbCharacterEncoding = dbCharacterEncoding;
21        this.createConnection();
22    }
23    private void createConnection() throws ClassNotFoundException, SQLException {
24        Class.forName(this.dbDriver);
25        Properties info = new Properties();
26        info.setProperty("characterEncoding", this.dbCharacterEncoding);
27        info.setProperty("user", this.dbUser);
28        info.setProperty("password", this.dbPassword);
29        this.connection = DriverManager.getConnection(this.dbURL, info);
30    }
31    public Connection getConnection() {
32        return this.connection;
33    }
34
35    public void closeConnection() throws SQLException {
36        this.connection.close();
37    }
38 }
```

# Bước 0: Xây dựng DBConnectionManager

```
*web.xml
1 <?xml version="1.0" encoding="UTF-8"?>
2 <web-app xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns="http://xmlns.jcp.org/xml/n:
3   <display-name>listener-sample-1</display-name>
4   <welcome-file-list>
5     <welcome-file>index.do</welcome-file>
6   </welcome-file-list>
7   <context-param>
8     <param-name>dbDriver</param-name>
9     <param-value>org.postgresql.Driver</param-value>
10  </context-param>
11  <context-param>
12    <param-name>dbUser</param-name>
13    <param-value>nhanh@fit.hcmus.edu.vn</param-value>
14  </context-param>
15  <context-param>
16    <param-name>dbPassword</param-name>
17    <param-value>123456</param-value>
18  </context-param>
19  <context-param>
20    <param-name>dbURL</param-name>
21    <param-value>jdbc:postgresql://localhost:5432/BookOnline</param-value>
22  </context-param>
23  <context-param>
24    <param-name>dbCharacterEncoding</param-name>
25    <param-value>UTF-8</param-value>
26  </context-param>
27 </web-app>
```

# Bước 1: Cài đặt listener

```
*AppContextListener.java
1 package org.nhanh.listener;
2 import java.sql.SQLException;
10 public class AppContextListener implements ServletContextListener {
11
12     public void contextInitialized(ServletContextEvent event) {
13         ServletContext ctx = event.getServletContext();
14         String dbUser = ctx.getInitParameter("dbUser");
15         String dbPassword = ctx.getInitParameter("dbPassword");
16         String dbURL = ctx.getInitParameter("dbURL");
17         String dbCharacterEncoding = ctx
18             .getInitParameter("dbCharacterEncoding");
19         String dbDriver = ctx.getInitParameter("dbDriver");
20         try {
21             DBConnectionManager dbManager = new DBConnectionManager(dbDriver,
22                 dbUser, dbPassword, dbURL, dbCharacterEncoding);
23             ctx.setAttribute("DBManager", dbManager);
24             System.out.println("Database connection initialized for Application.");
25         } catch (ClassNotFoundException | SQLException e) {
26             e.printStackTrace();
27         }
28     }
29     public void contextDestroyed(ServletContextEvent event) {
30         ServletContext ctx = event.getServletContext();
31         DBConnectionManager dbManager = (DBConnectionManager) ctx
32             .getAttribute("DBManager");
33         try {
34             dbManager.closeConnection();
35         } catch (SQLException e) {
36             e.printStackTrace();
37         }
38         System.out.println("Database connection closed for Application.");
39     }
40 }
```



## Bước 2: Đăng ký listener

- (1) Khai báo bằng **Annotation**

**@WebListener**

```
public class AppContextListener implements ServletContextListener
```

- (2) Khai báo trong **web.xml**

```
<web-app ...>
  <display-name>application-context-listener-sample</display-
name>
  <listener>
    <listener-class>
      org.nhanh.listener.AppContextListener
    </listener-class>
  </listener>
</web-app>
```

## Bước 3: Sử dụng các attribute

- **Controller** (Servlet)

```
ServletContext app = request.getServletContext();
```

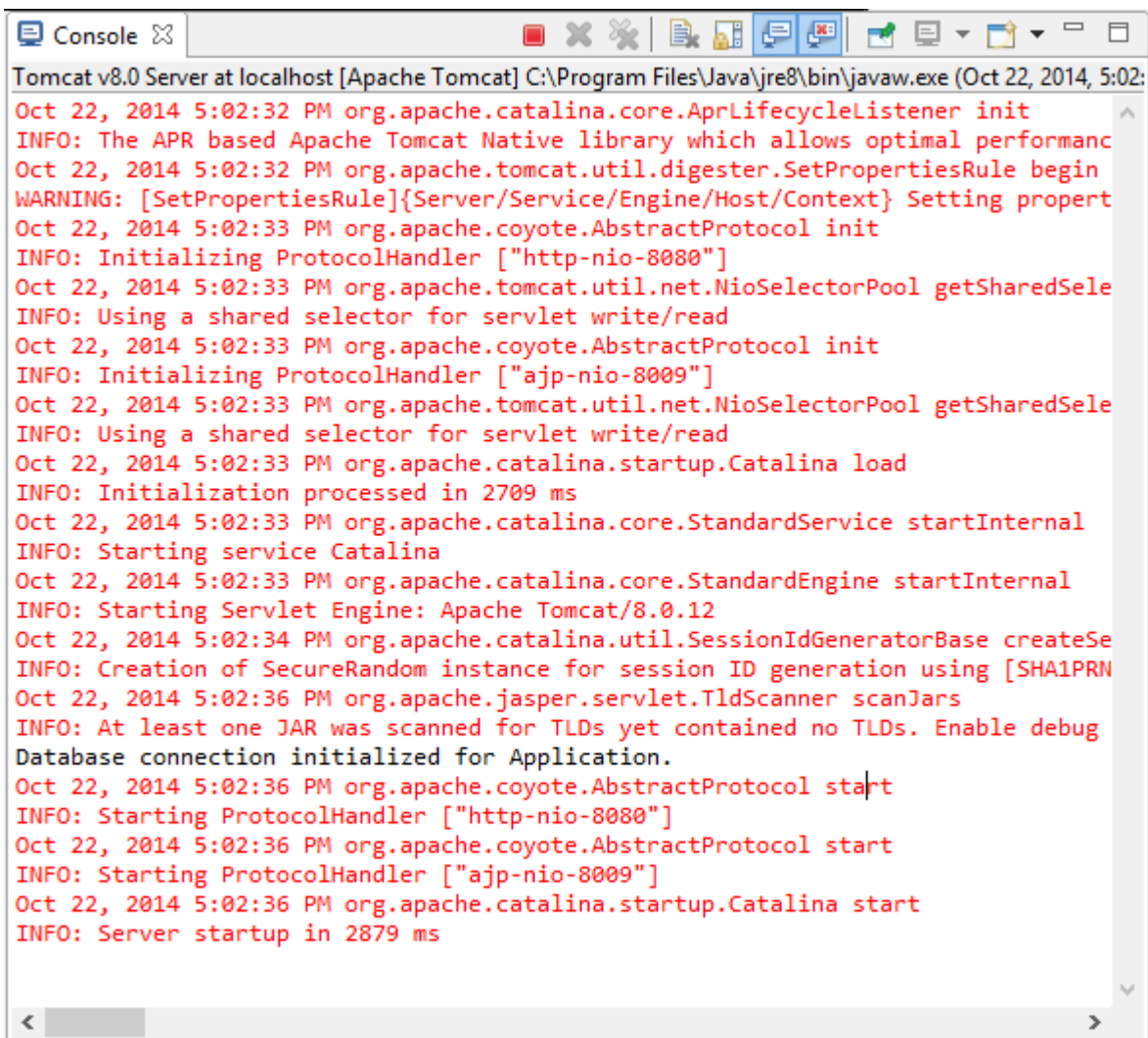
```
DBConnectionManager db =
```

```
    (DBConnectionManager)app.getAttribute("DBManager");
```

```
db...
```

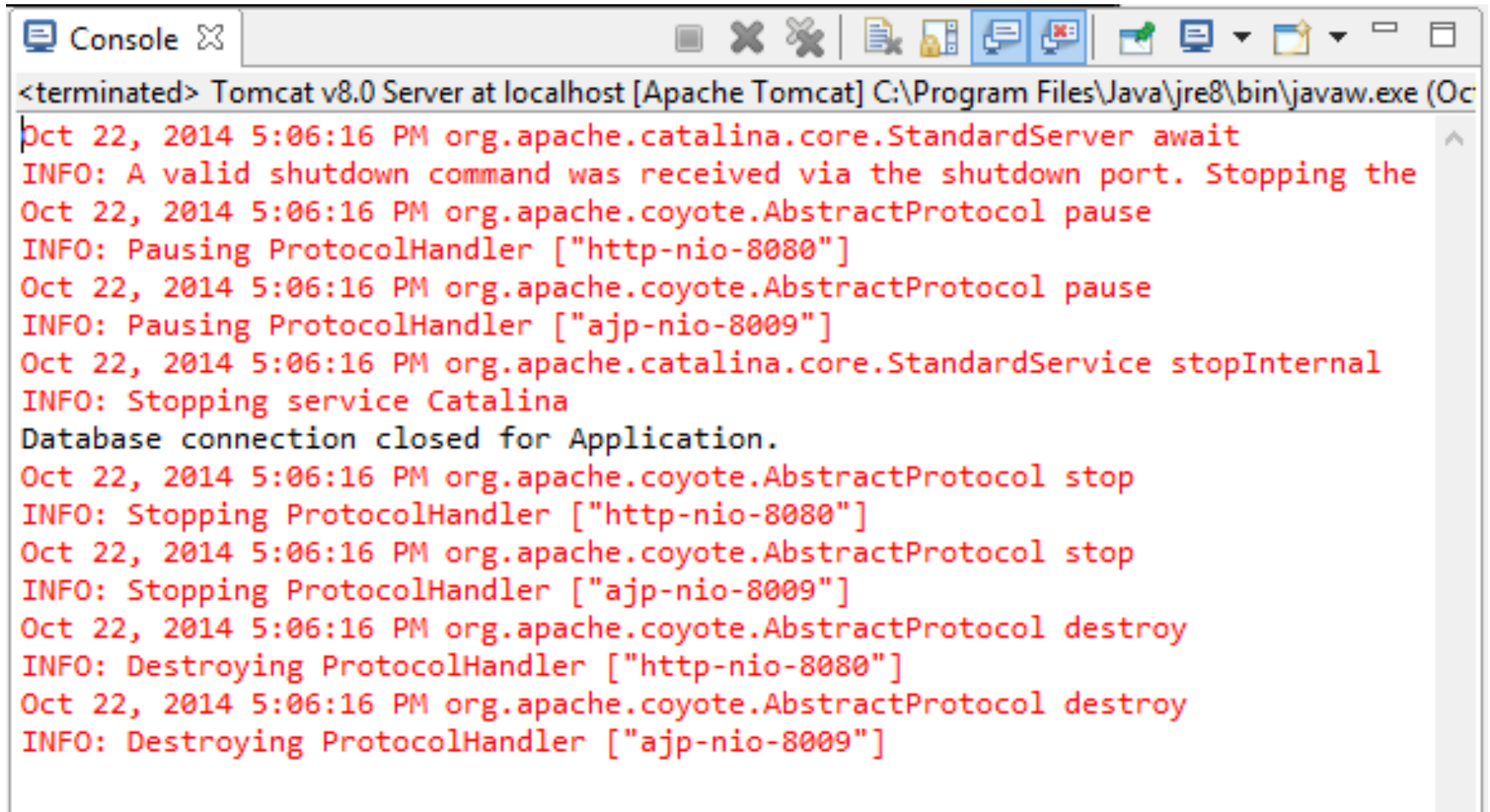
```
.....
```

# Event : Web Application được start

A screenshot of a Java console window titled "Console". The window shows the startup logs of an Apache Tomcat v8.0 Server at localhost. The logs are displayed in a monospaced font with red text on a white background. The logs include timestamps, class names, and messages such as "init", "begin", "WARNING", "INFO", and "start". The console window has a standard Windows-style title bar and a toolbar with icons for various actions like copy, paste, and search.

```
Tomcat v8.0 Server at localhost [Apache Tomcat] C:\Program Files\Java\jre8\bin\javaw.exe (Oct 22, 2014, 5:02:32 PM)
Oct 22, 2014 5:02:32 PM org.apache.catalina.core.AprLifecycleListener init
INFO: The APR based Apache Tomcat Native library which allows optimal performance
Oct 22, 2014 5:02:32 PM org.apache.tomcat.util.digester.SetPropertiesRule begin
WARNING: [SetPropertiesRule]{Server/Service/Engine/Host/Context} Setting property
Oct 22, 2014 5:02:33 PM org.apache.coyote.AbstractProtocol init
INFO: Initializing ProtocolHandler ["http-nio-8080"]
Oct 22, 2014 5:02:33 PM org.apache.tomcat.util.net.NioSelectorPool getSharedSelector
INFO: Using a shared selector for servlet write/read
Oct 22, 2014 5:02:33 PM org.apache.coyote.AbstractProtocol init
INFO: Initializing ProtocolHandler ["ajp-nio-8009"]
Oct 22, 2014 5:02:33 PM org.apache.tomcat.util.net.NioSelectorPool getSharedSelector
INFO: Using a shared selector for servlet write/read
Oct 22, 2014 5:02:33 PM org.apache.catalina.startup.Catalina load
INFO: Initialization processed in 2709 ms
Oct 22, 2014 5:02:33 PM org.apache.catalina.core.StandardService startInternal
INFO: Starting service Catalina
Oct 22, 2014 5:02:33 PM org.apache.catalina.core.StandardEngine startInternal
INFO: Starting Servlet Engine: Apache Tomcat/8.0.12
Oct 22, 2014 5:02:34 PM org.apache.catalina.util.SessionIdGeneratorBase createSecureRandom
INFO: Creation of SecureRandom instance for session ID generation using [SHA1PRN]
Oct 22, 2014 5:02:36 PM org.apache.jasper.servlet.TldScanner scanJars
INFO: At least one JAR was scanned for TLDs yet contained no TLDs. Enable debug logging for this logger to see the full JAR scanning results.
Database connection initialized for Application.
Oct 22, 2014 5:02:36 PM org.apache.coyote.AbstractProtocol start
INFO: Starting ProtocolHandler ["http-nio-8080"]
Oct 22, 2014 5:02:36 PM org.apache.coyote.AbstractProtocol start
INFO: Starting ProtocolHandler ["ajp-nio-8009"]
Oct 22, 2014 5:02:36 PM org.apache.catalina.startup.Catalina start
INFO: Server startup in 2879 ms
```

# Event : Web Application được stop



```
<terminated> Tomcat v8.0 Server at localhost [Apache Tomcat] C:\Program Files\Java\jre8\bin\javaw.exe (Oc
Oct 22, 2014 5:06:16 PM org.apache.catalina.core.StandardServer await
INFO: A valid shutdown command was received via the shutdown port. Stopping the
Oct 22, 2014 5:06:16 PM org.apache.coyote.AbstractProtocol pause
INFO: Pausing ProtocolHandler ["http-nio-8080"]
Oct 22, 2014 5:06:16 PM org.apache.coyote.AbstractProtocol pause
INFO: Pausing ProtocolHandler ["ajp-nio-8009"]
Oct 22, 2014 5:06:16 PM org.apache.catalina.core.StandardService stopInternal
INFO: Stopping service Catalina
Database connection closed for Application.
Oct 22, 2014 5:06:16 PM org.apache.coyote.AbstractProtocol stop
INFO: Stopping ProtocolHandler ["http-nio-8080"]
Oct 22, 2014 5:06:16 PM org.apache.coyote.AbstractProtocol stop
INFO: Stopping ProtocolHandler ["ajp-nio-8009"]
Oct 22, 2014 5:06:16 PM org.apache.coyote.AbstractProtocol destroy
INFO: Destroying ProtocolHandler ["http-nio-8080"]
Oct 22, 2014 5:06:16 PM org.apache.coyote.AbstractProtocol destroy
INFO: Destroying ProtocolHandler ["ajp-nio-8009"]
```

# Lắng nghe các event ở application scope

- **Lắng nghe 2 event** liên quan đến **web application** được **start** hoặc **stop**:
  - `javax.servlet.ServletContextListener`
- **Lắng nghe các event** liên quan đến sự **thay đổi** các **attribute** trong **application scope**:
  - `javax.servlet.ServletContextAttributeListener`

# Lắng nghe các event ở application scope

- **Lắng nghe các event** liên quan đến sự **thay đổi** các **attribute** trong **application scope**:
  - `javax.servlet.ServletContextAttributeListener`

```
1 public interface implements ServletContextAttributeListener {  
2  
3     public void attributeAdded (ServletContextAttributeEvent event);  
4  
5     public void attributeReplaced (ServletContextAttributeEvent event) ;  
6  
7     public void attributeRemoved (ServletContextAttributeEvent event) ;  
8  
9 }
```

# ServletContextAttributeListener

- **Bước 1:** Cài đặt listener.
- **Bước 2:** Đăng ký listener.

# Bước 1: Cài đặt listener

```
*AppContextAttributeListener.java
1 package org.nhanh.listener;
2
3 import javax.servlet.ServletContextAttributeEvent;
4
5
6
7 @WebListener
8 public class AppContextAttributeListener implements ServletContextAttributeListener {
9
10     public void attributeAdded(ServletContextAttributeEvent e) {
11         System.out.println("ServletContext attribute added::{"+e.getName()+","+e.getValue()+"}");
12     }
13
14     public void attributeReplaced(ServletContextAttributeEvent e) {
15         System.out.println("ServletContext attribute replaced::{"+e.getName()+","+e.getValue()+"}");
16     }
17
18     public void attributeRemoved(ServletContextAttributeEvent e) {
19         System.out.println("ServletContext attribute removed::{"+e.getName()+","+e.getValue()+"}");
20     }
21
22 }
```



## Bước 2: Đăng ký listener

- (1) Khai báo bằng **Annotation**

**@WebListener**

```
public class AppContextAttributeListener  
    implements ServletContextListener
```

- (2) Khai báo trong **web.xml**

```
<web-app ...>  
    <display-name>application-context-listener-sample</display-  
name>  
    <listener>  
        <listener-class>  
            org.nhanh.listener.AppContextAttributeListener  
        </listener-class>  
    </listener>  
</web-app>
```

# ServletContextAttributeListener

- ServletContext attribute added::`{User,nhanh}`
- ServletContext attribute replaced::`{User,Pankaj}`
- ServletContext attribute removed::`{User, Pankaj}`

# Demo



# LẮNG NGHE CÁC EVENT Ở SESSION SCOPE

# Lắng nghe các event ở session scope

- **Lắng nghe 2 event** liên quan đến **session** được **create** hoặc **destroy**:
  - **javax.servlet.http.HttpSessionListener**

```
1 public interface HttpSessionListener {  
2  
3     public void sessionCreated(HttpSessionEvent event);  
4  
5     public void sessionDestroyed(HttpSessionEvent event);  
6 }
```

# Lắng nghe các event ở session scope

- **Lắng nghe các event** liên quan đến sự **thay đổi** các **attribute** trong **session scope** :
  - `javax.servlet.HttpSessionAttributeListener`

```
1 public interface HttpSessionAttributeListener {  
2  
3     public void attributeAdded(HttpSessionBindingEvent e);  
4  
5     public void attributeRemoved(HttpSessionBindingEvent e);  
6  
7     public void attributeReplaced(HttpSessionBindingEvent e);  
8  
9 }
```

# Lắng nghe các event ở session scope

- **Lắng nghe các event** liên quan đến sự **thay đổi** các **attribute** trong **session scope** của các **thể hiện** của **class** cài đặt từ:
  - `javax.servlet. HttpSessionBindingListener`

```
1 public interface HttpSessionBindingListener{
2
3     public void valueBound (HttpSessionBindingEvent event)
4
5     public void valueUnBound (HttpSessionBindingEvent event)
6
7 }
```

- **Ứng dụng?**

# Lắng nghe các event ở session scope

- Sau khi gọi **HttpSession#setAttribute**
  - **valueBound()** sẽ được gọi.
- Sau khi gọi
  - **HttpSession#removeAttribute()**
  - Hoặc **HttpSession#invalidate()**
  - Hoặc **attribute** được cập nhật bởi **HttpSession#setAttribute()**
- **valueUnbound()** sẽ được gọi.
- Lưu ý: chỉ liên quan đến các thể hiện của class cài đặt từ **HttpSessionBindingListener**



# Demo



# LẮNG NGHE CÁC EVENT Ở REQUEST SCOPE

# Lắng nghe các event ở request scope

- Lắng nghe 2 event liên quan đến request được create hoặc destroy:
  - `javax.servlet.http.HttpServletRequestListener`

```
1 public interface HttpServletRequestListener {  
2  
3     public void requestInitialized(ServletRequestEvent event);  
4  
5     public void requestDestroyed(ServletRequestEvent event);  
6 }
```

# Lắng nghe các event ở request scope

- Lắng nghe các event liên quan đến sự thay đổi các attribute trong request scope :
  - `javax.servlet.ServletRequestAttributeListener`

```
1 public interface ServletRequestAttributeListener{
2
3     public void attributeAdded(ServletRequestAttributeEvent e);
4
5     public void attributeRemoved(ServletRequestAttributeEvent e);
6
7     public void attributeReplaced(ServletRequestAttributeEvent e);
8
9 }
```

# Demo





# HỎI VÀ ĐÁP

# Reference

- [1] Joel Murach; Michael Urban. (2014). Murach's Java Servlets and JSP, 3<sup>rd</sup> Edition. Mike Murach & Associate.