COMP S380F Web Applications: Design and Development
Lab 5: JSP and Session activity tracking

In this lab, we will write a web application to track session activities. The following topics will be covered:

- Extracting information about the session object
- Session attributes
- The annotation @SuppressWarnings("unchecked")
- Java synchronized collection class: CopyOnWriteArrayList
- Configuring JSP properties in web.xml
- JSTL core tag: <c:redirect>

Task: We will create a servlet to track the visited URLs in a session, and a JSP page to display the tracked activities.

1. In IntelliJ, create a **Gradle Web Application** project with the following properties:
    - Category: **Jakarta EE**
    - Name: **Lab05**
    - Template: **Web application**
    - Application Server: **Tomcat 10.1**
    - Build system: **Gradle**
    - Group: **hkmu.comps380f**
    - Jakarta EE Version: **Jakarta EE 10**

2. In "Project" section, select **Lab05 > build.gradle**. Update `dependencies` to add JSTL (with an API and an implementation), as follows. Then, it is important to *reload the project in the Gradle tool window* (on the right-side menu) to let Gradle download the required libraries from the Maven central repository.

```
dependencies {
    compileOnly('jakarta.servlet:jakarta.servlet-api:6.0.0')
    implementation 'jakarta.servlet.jsp.jstl:jakarta.servlet.jsp.jstl-api:3.0.0'
    implementation 'org.glassfish.web:jakarta.servlet.jsp.jstl:3.0.1'
}
```

You can also search and generate a Gradle dependency, as shown in the following link:
https://www.jetbrains.com/help/idea/work-with-gradle-dependency-diagram.html#gradle_generate

3. Add the following <jsp-config> to the DD (/WEB-INF/web.xml):

```
<jsp-config>
    <jsp-property-group>
        <url-pattern>*.jsp</url-pattern>
        <url-pattern>*.jspf</url-pattern>
        <page-encoding>UTF-8</page-encoding>
        <include-prelude>/WEB-INF/jsp/base.jspf</include-prelude>
        <trim-directive-whitespaces>true</trim-directive-whitespaces>
        <default-content-type>text/html</default-content-type>
    </jsp-property-group>
</jsp-config>
```

Since we will use the JSTL core tags in our JSP pages, the JSP fragment `/WEB-INF/jsp/`**`base.jspf`** contains the following tag that declares the JSTL core tag library with an XMLNS prefix "c". This will be included in all JSPs in the web app; the web container is smart enough not to include it to `base.jspf` itself.

```
<%@taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>
```

4. In the Java package **hkmu.comps380f**, create a **JavaBean** PageVisit. A PageVisit object will record a single page visit activity, and has the following properties:
    - long enteredTimestamp: the time when the page is visited
    - Long leftTimestamp: the time when the page is left. This variable is a Long object instead of a primitive long value, so it allows null value.
    - String request: the URL of the visited page
    - InetAddress ipAddress: the IP address of the client

5. Create the Servlet ActivityServlet that contains the business logic for tracking the session activities.

```
@WebServlet(
        name = "activityServlet",
        urlPatterns = {"/do/*"}
)
public class ActivityServlet extends HttpServlet {
    @Override
    protected void doGet(HttpServletRequest request, HttpServletResponse response)
            throws ServletException, IOException {
        this.recordSessionActivity(request);
        this.viewSessionActivity(request, response);
    }

    // Defining other methods ...
}
```

6. **recordSessionActivity()**: This method keeps the current page visit in a PageVisit object and adds it to **a list of PageVisit objects**; this list is stored as a **session attribute**.

- For the list, we will use CopyOnWriteArrayList which is synchronized (i.e., thread-safe), instead of ArrayList which is not synchronized.
- The cast to a **CopyOnWriteArrayList<PageVisit>** is an unchecked operation and will create a warning. To suppress the warning, we use the annotation @SuppressWarnings("unchecked").

```
private void recordSessionActivity(HttpServletRequest request) {
    HttpSession session = request.getSession();
    if (session.getAttribute("activity") == null)
        session.setAttribute("activity", new CopyOnWriteArrayList<PageVisit>());

    @SuppressWarnings("unchecked")
    CopyOnWriteArrayList<PageVisit> visits
            = (CopyOnWriteArrayList<PageVisit>) session.getAttribute("activity");
    if (!visits.isEmpty()) {
        PageVisit last = visits.get(visits.size()-1);
        last.setLeftTimestamp(System.currentTimeMillis());
    }

    PageVisit now = new PageVisit();
    now.setEnteredTimestamp(System.currentTimeMillis());
    if (request.getQueryString() == null) now.setRequest(request.getRequestURL().toString());
    else now.setRequest(request.getRequestURL() + "?" + request.getQueryString());
    try {
        now.setIpAddress(InetAddress.getByName(request.getRemoteAddr()));
    } catch (UnknownHostException e) {
        e.printStackTrace();
    }
    visits.add(now);
}
```

7. **viewSessionActivity()**: This method shows the list of session activities by forwarding to the JSP presentation page in **/WEB-INF/jsp/viewSessionActivity.jsp**.

   Complete this method by yourself. This function may throw **ServletException** and **IOException**.

8. Create the following JSP page `viewSessionActivity.jsp`:

```
<%@ page import="java.util.concurrent.CopyOnWriteArrayList, hkmu.comps380f.PageVisit" %>
<%@ page import="java.util.Date, java.text.SimpleDateFormat" %>
<%!
    private static String toString(long timeInterval) {
        if (timeInterval < 1_000)
            return "less than one second";
        if (timeInterval < 60_000)
            return (timeInterval / 1_000) + " seconds";
        return "about " + (timeInterval / 60_000) + " minutes";
    }
%>
<%
    SimpleDateFormat f = new SimpleDateFormat("EEE, d MMM yyyy HH:mm:ss Z");
%>
<!DOCTYPE html>
<html>
<head><title>Session Activity</title></head>
<body>
<h1>Session Activity</h1>

<h2>Session properties</h2>
Session ID: <%= session.getId() %><br/>
Session is new: <%= session.isNew() %><br/>
Session created: <%= f.format(new Date(session.getCreationTime())) %><br/>

<h2>Page activity in this session</h2>
<%
    @SuppressWarnings("unchecked")
    CopyOnWriteArrayList<PageVisit> visits =
            (CopyOnWriteArrayList<PageVisit>) session.getAttribute("activity");

    for (PageVisit visit : visits) {
        out.print(visit.getRequest());
        if (visit.getIpAddress() != null)
            out.print(" from IP "
                    + visit.getIpAddress().getHostAddress());
        out.print(" (" + f.format(new Date(visit.getEnteredTimestamp())));
        if (visit.getLeftTimestamp() != null) {
            out.print(", stayed for " + toString(
                    visit.getLeftTimestamp() - visit.getEnteredTimestamp()
            ));
        }
        out.println(")<br />");
    }
%>
</body>
</html>
```

9. Update `index.jsp` to contain the following single line, where the JSTL core tag **`<c:redirect>`** redirects the user to another URL:

```
<c:redirect url="/do/home" />
```

10. Build and deploy the web app. Access the `index.jsp` page and then access the URL `/do/*` with different strings for `*`. You may also append different query parameters to the URLs.

11. Use another browser to browse the page `/do/home`. What is the Boolean value in the line "Session is new"? Is it equal to the Boolean value shown in Step 10?