Tehnologii Java pentru aplicații distribuite - Tema 1 - Servlet

Student: Stoia Alin

Grupa: 244

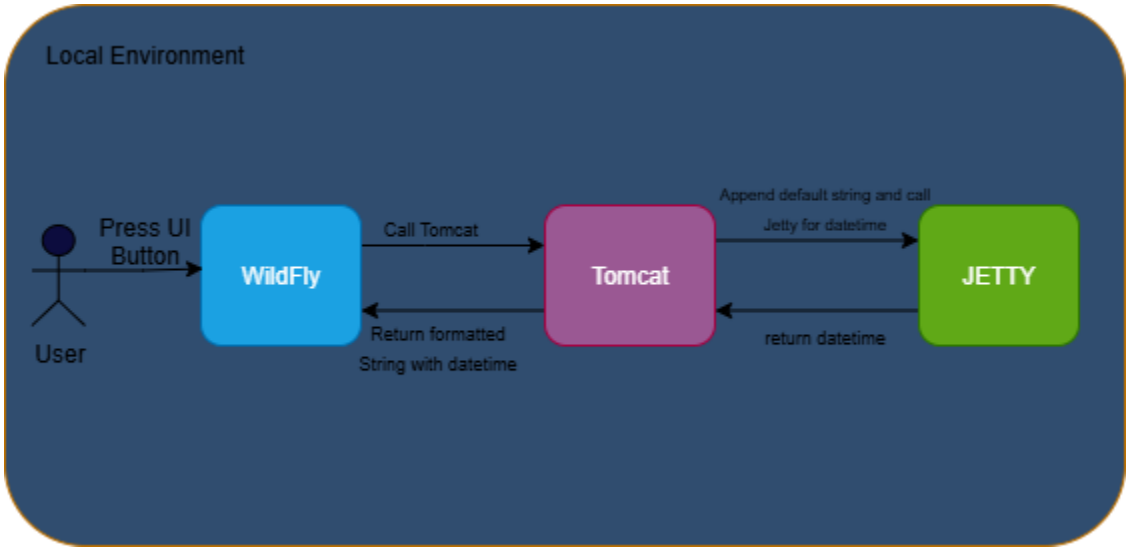
Introducere

Lucrarea de față prezintă implementarea a trei aplicații Java pe servere diferite de tip Servlet Container (Tomcat, Jetty și WildFly). Scopul lucrării este de a exemplifica mecanismele de expunere a endpoint-urilor prin Servlets, respectiv comunicarea între aplicații independente.

Arhitectura sistemului

Sistemul este format din trei aplicații independente, care comunică între ele:
1. TomcatServletApp – expune endpointul /hello și apelează Jetty pentru a prelua data și ora curentă.
2. JettyServletApp – expune endpointul /datetime și returnează data și ora curentă.
3. WildflyEJBApp – aplicație Java EE care apelează Tomcat și prin el Jetty, afișând rezultatul într-o pagină JSP.

Fluxul arhitectural este următorul: Client → WildFly → Tomcat → Jetty.



Prerequisites și Setup pentru cele 3 servere

Pentru realizarea aplicațiilor Java distribuite, proiectul utilizează trei servere diferite: **Tomcat**, **Jetty** și **WildFly**. Înainte de a rula aplicațiile, trebuie respectate următoarele:

- **IDE:** IntelliJ IDEA
- **Java:** JDK 8 (exemplu: Corretto 1.8.0_462)
- **Maven:** versiunea 3.x
- **Servere:**
  - Apache Tomcat 9.0.109

    - Jetty-Distribution 9.4.58.v20250814
    - WildFly 19.1.0.Final (compatibil cu Java EE 8)
- Setarea variabilelor de mediu: JAVA_HOME și WILDFLY_HOME.

Descrierea componentelor și a metodelor

HelloServlet (Tomcat)
Metode expuse:

- protected void doGet(HttpServletRequest req, HttpServletResponse resp)
- public String callJetty(String urlString)

Rol: primește cererea, încearcă să obțină data și ora din Jetty. Excepții posibile: IOException - conexiune refuzată.

Setup:

- Crearea proiectului Maven în IntelliJ IDEA, tip *maven-archetype-webapp*.
- Configurarea dependinței jakarta.servlet-api cu <scope>provided</scope> în pom.xml.
- Crearea directorului java ca *Sources Root* și pachetul com.sa.tpjad.tomcat pentru clasa HelloServlet.
- Compilarea proiectului cu mvn clean install pentru generarea WAR-ului (TomcatServletApp.war).
- Copierea WAR-ului în directorul webapps al Tomcat și pornirea serverului (start.bat).

DateTimeServlet (Jetty)

Metode expuse:

- protected void doGet(HttpServletRequest req, HttpServletResponse resp)

Rol: returnează data și ora curentă formatată.

Setup:

- Crearea proiectului Maven în IntelliJ IDEA, tip *maven-archetype-webapp*.
- Configurarea dependinței jakarta.servlet-api cu <scope>provided</scope> în pom.xml.
- Definirea servlet-ului în web.xml (în cazul de față fără adnotări).

- Crearea unui Jetty base (jetty_tpjad_base) și plasarea WAR-ului în directorul webapps.
- Configurarea fișierului start.ini cu modulele necesare (deploy, http, server).
- Pornirea serverului:

```
cd jetty_tpjad_base
java -Djetty.port=8081 -jar ../start.jar
```

ButtonServlet (WildFly)

Metode expuse:

- protected void doGet(HttpServletRequest request, HttpServletResponse response)

Rol: face request către Tomcat și returnează rezultatul către client. Excepții posibile: IOException, conexiune nereușită.

Setup:

1. Crearea proiectului Maven în IntelliJ IDEA, tip *maven-archetype-webapp*.
2. Configurarea dependinței jakarta.jakartaee-api cu <scope>provided</scope> în pom.xml.
3. Compilarea proiectului și generarea WAR-ului (WildFlyEJBApp-1.1-SNAPSHOT.war).
4. Copierea WAR-ului în directorul standalone/deployments al WildFly.
5. Asigurarea că nu există deployment-uri vechi în standalone.xml.
6. Pornirea serverului:

```
cd %WILDFLY_HOME%\bin
standalone.bat -Djboss.http.port=8082
```

index.jsp (WildFly)

Rol: oferă interfață HTML cu un buton care, prin JavaScript, apelează servletul /butonCareCheamaTomcat.

Descrierea funcționalității

Succesiunea apelurilor este următoarea:

1. Utilizatorul apasă butonul „Ce zi e azi?” din pagina JSP (WildFly).
2. WildFly trimite request la Tomcat.
3. Tomcat apelează Jetty.
4. Jetty returnează data și ora
5. Tomcat formatează un String care va conține un text predefinit si rezultatul de la Jetty.
6. WildFly adaugă conținutul răspunsului în variabila ${message}

Dacă Jetty nu este pornit, Tomcat returnează mesajul default „Habar n-am”. Daca Tomcat nu este pornit, în JSP va fi afișat textul “Nu vom afla niciodata” urmat de excepția aruncată în urma apelului invalid.

Exemple de utilizare

- Jetty: http://localhost:8081/JettyServletApp/datetime pentru returnarea datei în formatul (“yyyy-MM-dd HH:mm:ss)

- Tomcat: http://localhost:8080/TomcatServletApp/hello pentru returnarea String-ului “Servus! Ora este: <data și ora>

- WildFly: http://localhost:8082/WildflyEJBApp pentru accesarea interfeței, implicit a fluxului end-to-end.