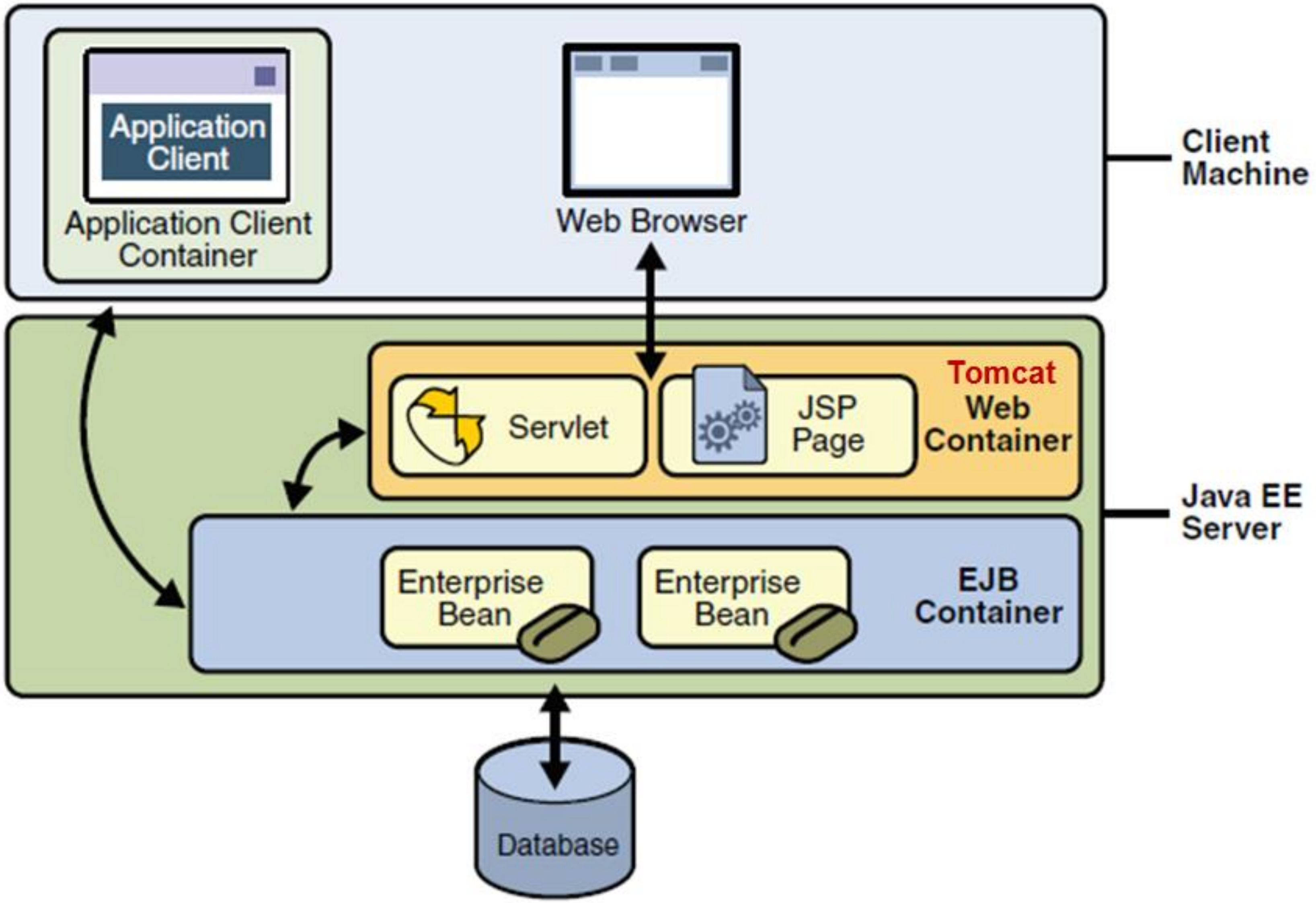
Servlets & JavaServer Pages (JSPs)

Дистрибуирани софтверски системи

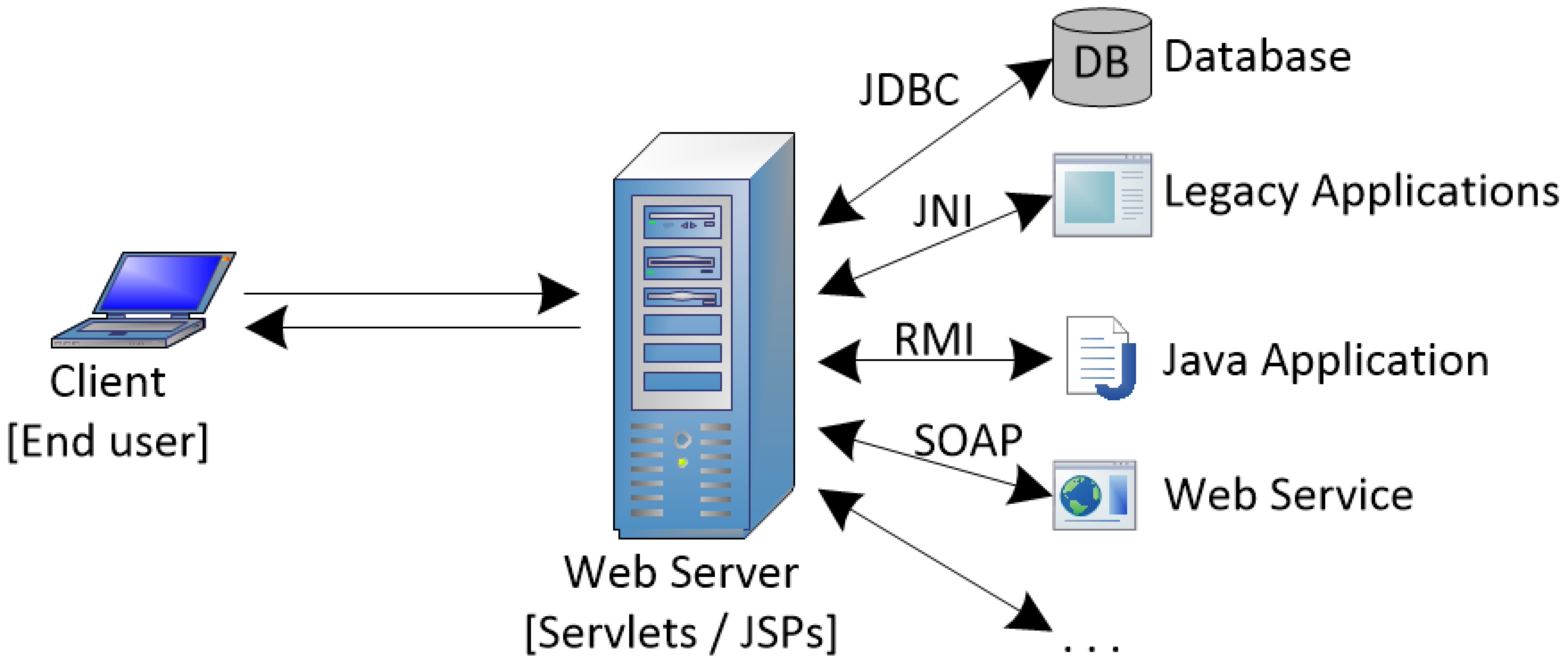
Предметни наставник: проф. др Жељко Стојанов

Универзитет у Новом Саду
Технички факултет “Михајло Пупин”

Servlets & JSPs in Web Containers



Web Middleware Role



Java Database Connectivity (JDBC) – API for accessing relational databases

Java Native Interface (JNI) – programming framework for communicating with native applications

Java Remote Method Invocation (Java RMI) – mechanism for distributed Java applications

Simple Object Access Protocol (SOAP) - XML-based messaging protocol

Servlets

Servlet

A servlet is a Java class used to extend the capabilities of web servers that host applications accessed via a client-server programming model.

https://javaee.github.io/tutorial/servlets.html

All servlets must implement the Servlet interface, which defines the life-cycle methods, or extend one of the classes:

- **GenericServlet –** handles generic services.
- **HttpServlet –** handles HTTP services, extends GenericServlet

Packages:

- **javax.servlet**
  https://docs.oracle.com/javaee/7/api/javax/servlet/package-summary.html
- **javax.servlet.http**
  https://docs.oracle.com/javaee/7/api/javax/servlet/http/package-summary.html

Servlet tasks

- **Read the explicit data sent by the client**
  User data entered by HTML web form or data from user application

- **Read the implicit HTTP request data sent by the browser**
  Behind-the-scenes HTTP information, which includes cookies, information about media types and compression schemes the browser understands, etc.

- **Generate the results**
  This process may require talking to a database, executing an RMI or EJB call, invoking a Web service, or computing the response directly.

- **Send the explicit data (i.e., the document) to the client**
  The most common format is HTML, so an important servlet/JSP task is to wrap the results inside of HTML. Data can be sent also as a document in a variety of formats, such as text (HTML or XML), binary (GIF images), or even a compressed format like gzip.

- **Send the implicit HTTP response data**
  HTTP response data involves telling the browser or other client what type of document is being returned (e.g., HTML), setting cookies and caching parameters, and other such tasks.

Advantages of using Servlets

- Efficiency
  - Single instance of a servlet and one thread per request
  - Servlet maintains it state on the server between the requests
  - Requests are handled via servlet methods
- Utilities for performing typical server tasks
  - Logging
  - Error management
  - Session management etc.
- Communication
  - Standardised way of communicating with the server
  - Servlets can share data
- Advantages of Java
  - Large number of APIs: JDBC, threads, RMI, networks, etc.
  - Portability between platforms and servers
  - Security
  - Object-oriented
  - Large community of developers
  - External code easily used

Servlet Life-Cycle

- **Instantiation & initialisation (on the first request)**
  - If no instance of servlet exists, the web container loads the servlet class, creates an instance, and initialises the instance by calling the servlet's `init` method

- **Handling of subsequent requests**
  - Container creates new thread that calls the `service` method of the instance
  - The `service` method determines what type of request has arrived and calls the appropriate method.

- **Destruction**
  - When the container decides to remove a servlet from the server, it first calls its `destroy` method

Servlet Life-Cycle Consequences

- Single virtual machine
  - Servlets can share data
- Persistence (in memory) of servlets' instances
  - Reduced memory consumption
  - Elimination of instantiation and initialisation time
  - Persistence (in memory) of state, data and resources
  - Persistence (in memory) of threads
- Concurrent requests
  - Synchronization of concurrent requests

Servlet configuration in web.xml

Servlets should be configured in Web deployment descriptor file **web.xml**

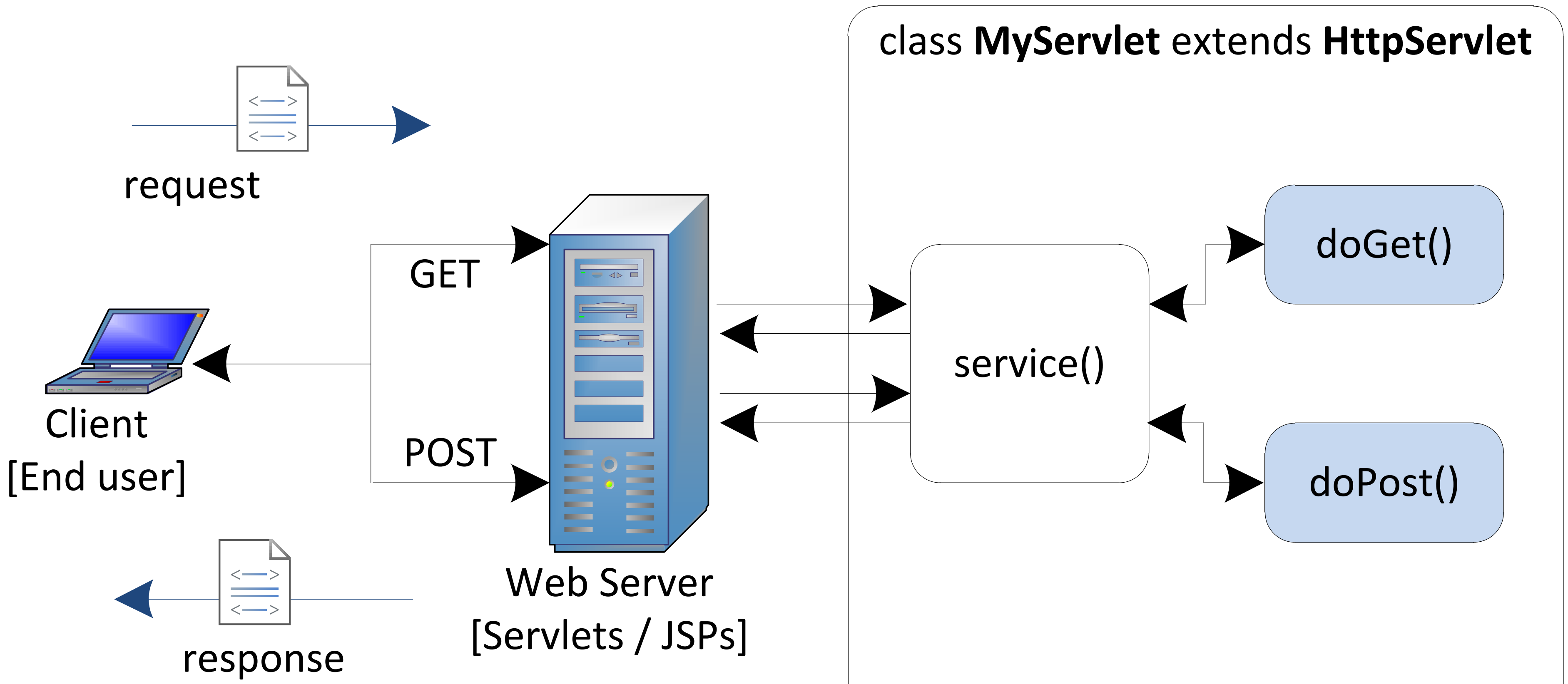
```
<web-app>
. . .

   <servlet>
      <servlet-name>UpdateCourseLecturer</servlet-name>
      <servlet-class>UpdateCourseLecturerServlet</servlet-class>

      <init-param>
         <param-name>lecturerEmail</param-name>
         <param-value>zeljko.stojanov@tfzr.rs</param-value>
      </init-param>
      . . .
      <init-param> . . . </init-param>
   </servlet>
. . .
</web-app>
```

Servlet mapping in web.xml

Servlet mapping specifies the web container of which java servlet should be invoked for a url given by client. It maps url patterns to servlets. When there is a request from a client, servlet container decides to which application it should forward to. Then context path of url is matched for mapping servlets.

```
<web-app>
. . .
    <servlet>
        <servlet-name>TextServlet</servlet-name>
        <servlet-class>TextServlet</servlet-class>
    . . .
    </servlet>
    . . .
    <servlet-mapping>
        <servlet-name>TextServlet</servlet-name>
        <url-pattern>/TextServlet</url-pattern>
    </servlet-mapping>
. . .
</web-app>
```

HttpServlet



JavaServer Pages (JSPs)

JavaServer Pages (JSPs)

JSP is java server-side technology for creating dynamic web pages by mixing static HTML content and dynamically created content. JSP is complimentary technology to Java Servlets.

JSPs enable clear separation of:

- Presentation written within HTML tags
- Business logic written in Java programming language inserted within special tags.
  It assumes use of Servlets, Java Beans, Enterprise Java Beans, etc.

Execution of JSPs

- JSP is translated into Servlet during the first access
- Servlet is loaded into memory and call for the next access to JSP
- `doGet` and `doPost` call `_jspService` method
- For initialization is called `jspInit`

JSP operation on web server

	JSP page translated into servlet	Servlet compiled	Servlet loaded into Server's Memory	`jspInit` called	`_jspService` called
JSP first written					
Request 1	Yes	Yes	Yes	Yes	Yes
Request 2	No	No	No	No	Yes
Server restarted					
Request 3	No	No	Yes	Yes	Yes
Request 4	No	No	No	No	Yes
Page modifies					
Request 5	Yes	Yes	Yes	Yes	Yes
Request 6	No	No	No	No	Yes

JSP basic syntax [1]

JSP Comment

Comments do not appear to user

```
<%-- Blah --%>
```

JSP Expression

Expressions are calculated and sent to user on each page request

```
<%= Java Value %>
```

JSP Scriptlet

Java statements executed on each page request

```
<% Java Statement %>
```

JSP basic syntax [2]

JSP Declaration

Field or method that become part of the class during translation to servlet

```
<%! Field Definition %>
<%! Method Definition %>
```

JSP Directive

Information of code structure of servlet (`page`), code included during compilation (`include`), or use of tag library (`taglib`)

```
<%@ directive att="val" %>
```

JSP Action

Action requested during page call

Options: `jsp:include`, `jsp:useBean`, `jsp:invoke`

```
<jsp:blah>...</jsp:blah>
```

Invoking dynamic code from JSP

Call Java code directly

All Java code is placed in JSP page. Appropriate only for very small chunks of code.

Call Java code indirectly

JSP contains only code necessary for invoking separate utility classes.

Use beans

JSP uses separate utility classes structured as beans.
Use `jsp:useBean`, `jsp:getProperty`, and `jsp:setProperty`.

Use the MVC architecture (the most complex applications)

Servlet responds to original request, look up data, and store results in beans.
After that it forward to a JSP page for presenting results.

Why limiting code amount in JSP

Development

- ❖ Java code is written in IDE (e.g. Eclipse).
- ❖ JSP is written in HTML editor.
- ❖ Different teams for Java development and Web development

Compilation

- ❖ Java code is build in IDE (e.g. Eclipse).
- ❖ JSP is compiled during the firs invoke at web server (e.g. Tomcat)

Debugging

- ❖ Java code is traced in IDE (e.g. Eclipse).
- ❖ JSP is traced at web server, which report errors (e.g. Tomcat)

Reuse

- ❖ Organization of project into frontend and backend
- ❖ Development of classes, libraries, modules by using OO principles