



JEE Course

Servlets - Introduction

Márcio Fuckner

HvA – Hogeschool van Amsterdam



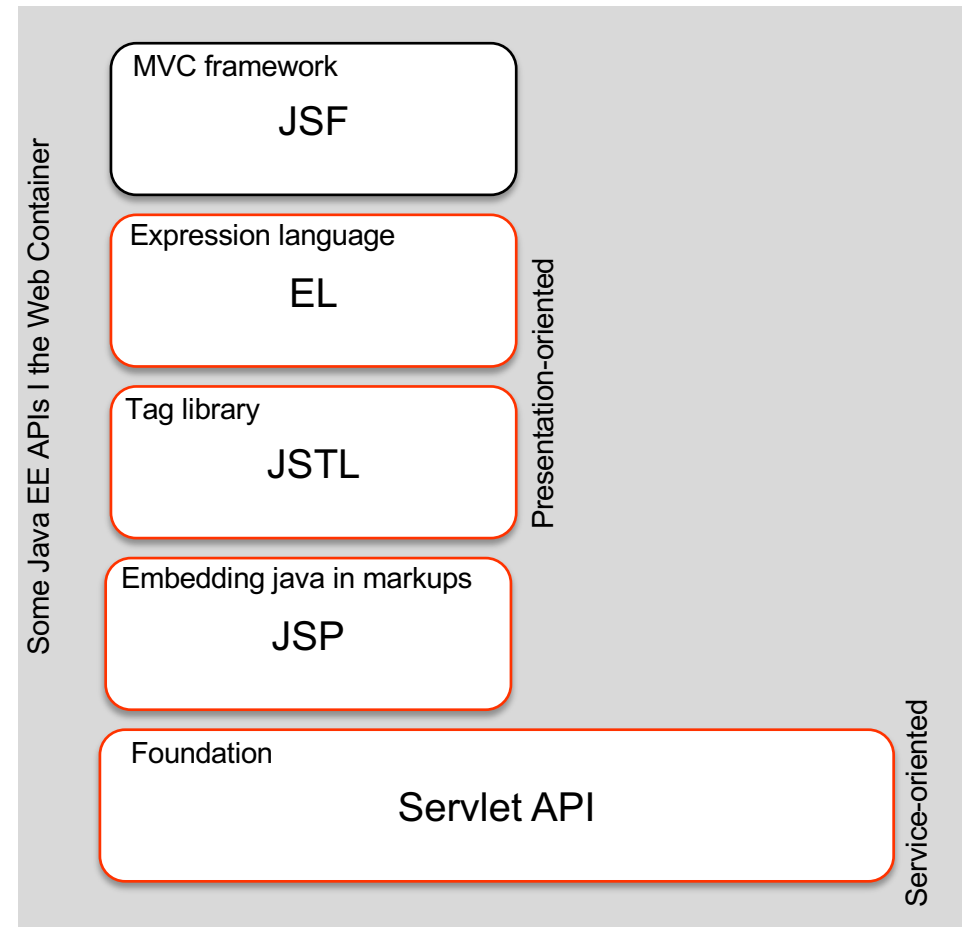
Content

- ▶ The web tier of the JEE ecosystem
- ▶ Web modules
- ▶ Servlet definition
- ▶ The request-response paradigm
- ▶ The servlet lifecycle



The web tier of the JEE ecosystem (1 de 2)

- **Servlet API**
 - Critical API used by several JEE APIs. Allows the execution of Java objects with server-side capabilities.
- **Java Server Pages API (JSP)**
 - A language entirely based on the Servlet API. Allows embedding Java code and custom tags into markup languages.
- **Java Server Pages Standard Tag Library (JSTL)**
 - Set of tags that helps to reduce the quantity of Java code in the presentation layer
- **Java Server Faces (JSF)**
 - Simplifies the creation of Web applications by providing a standard set of tools (or an API) for building user interfaces



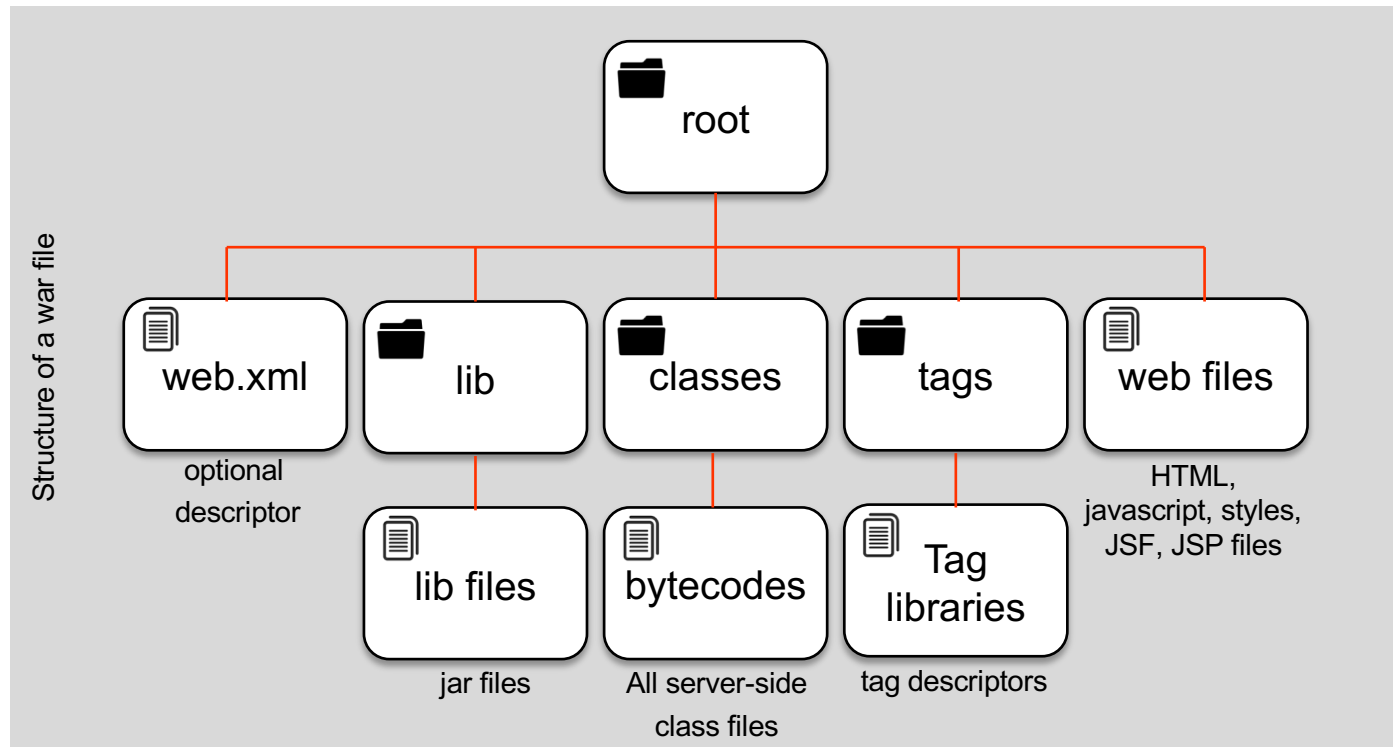


Web modules (1 of 2)

- ▶ A web module is the smallest deployable and functional unit of web resources. It contains web components and static web content files such as:
 - ▶ HTML pages, style files and JavaScript files.
 - ▶ Configuration descriptors
 - ▶ Custom tags;
 - ▶ Java classes
- ▶ A web module can be deployed as an unpacked file structure or can be packaged in a file known as a Web Archive (**WAR**)
- ▶ A **WAR** file has a predefined structure and can be deployed in any JEE-compatible server



Web modules (2 of 2)





Servlets – Introduction (1 of 3)

What is a Servlet according to Oracle ?

“A servlet is a Java programming language class used to extend the capabilities of servers that host applications accessed by means of a **request-response** programming model.”

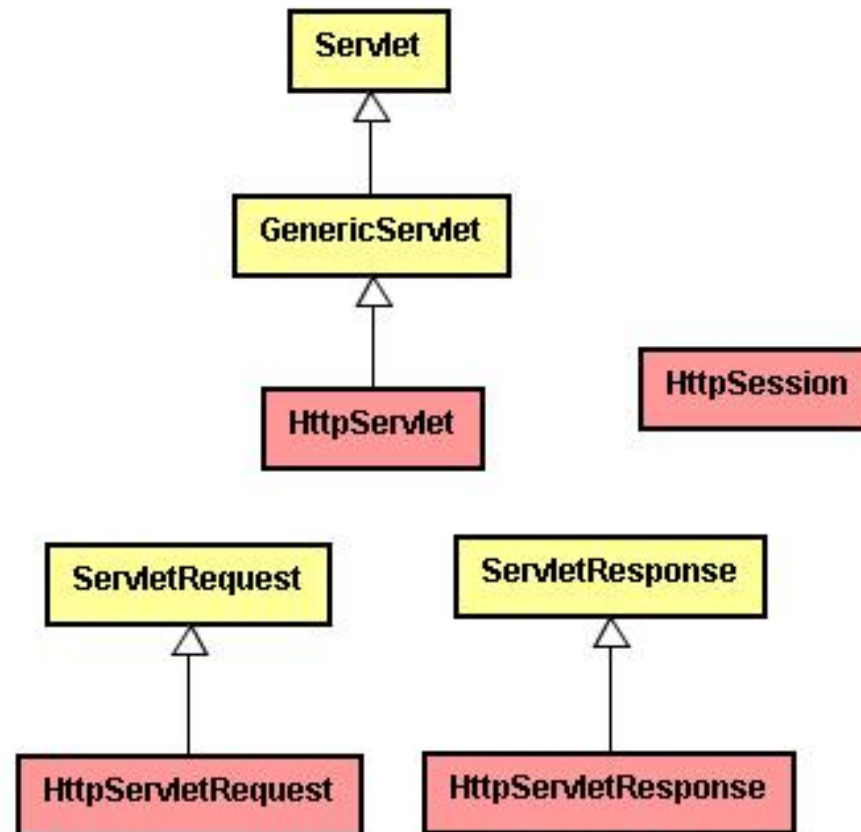


Servlets – Introduction (2 of 3)

- ▶ A servlet is a Java object that conforms to the Servlet API and inherits some functionalities of an HTTP server.
- ▶ These objects are accessible through **URL mappings**
- ▶ A servlet is **platform and server-independent**:
 - ▶ Servlets can be executed on any server compatible with the JEE specification



Servlets – Introduction (3 of 3)

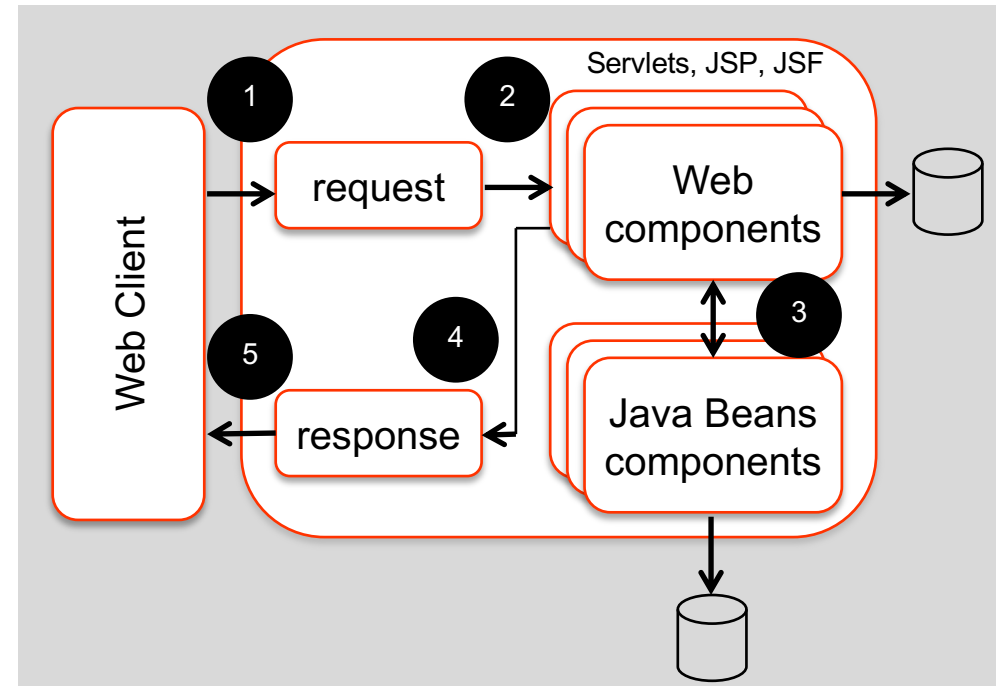


Basic Servlet API Hierarchy



Request-Response paradigm

- ▶ During the request, a typical servlet:
 - ▶ Receives HTTP requests
 - ▶ Extracts information such as parameters, attributes and cookies
 - ▶ Fires business logic (method calls, web service execution, EJB invocation, etc.)
- ▶ As a response, a servlet can:
 - ▶ Generate content (ex.: JSON, XML, HTML or binary)
 - ▶ Forwards the request to a proper component (another servlet, JSF or JSP page to render the content)



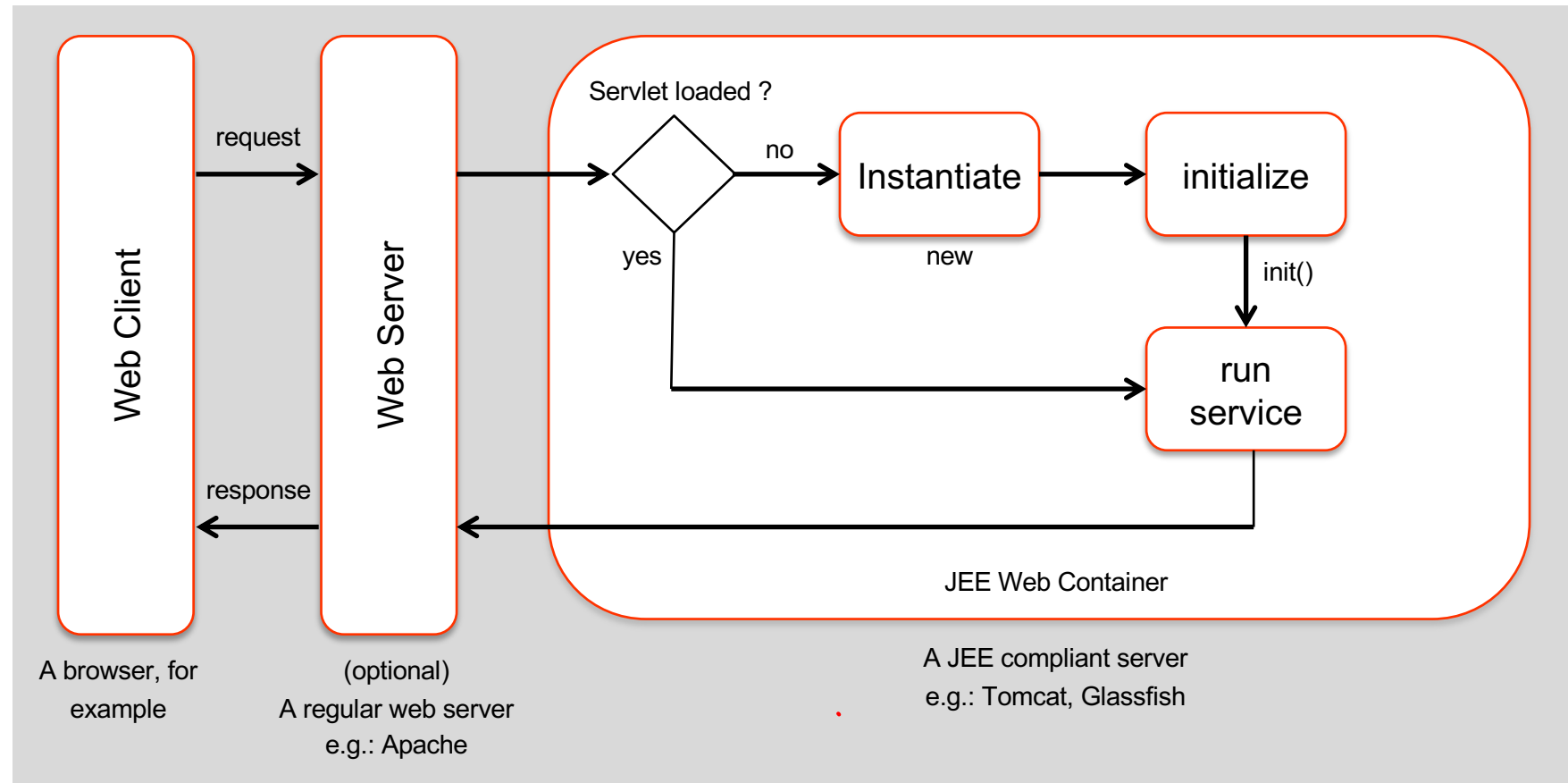


Servlet lifecycle (1 of 5)

- ▶ Servlets are naturally **multithreaded**
- ▶ A servlet instance can receive various requests at the same time
- ▶ The **container** controls the lifecycle of a servlet
- ▶ When the client invokes a servlet, the following steps are performed by the container
 - ▶ It **verifies** if an instance of the servlet exists
 - ▶ If not, it **creates** an instance of the servlet and executes the initialization methods
 - ▶ The container **invokes** the service method



Servlet lifecycle (2 of 5)



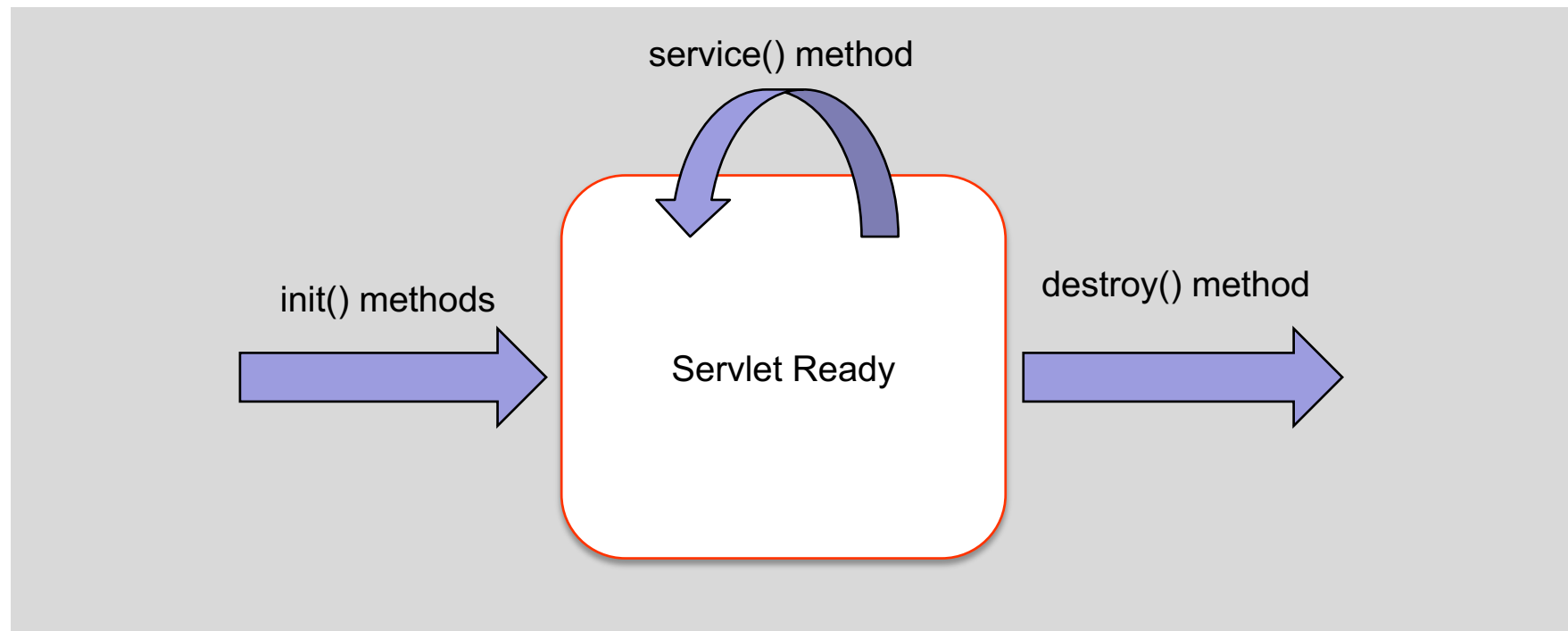


Servlet lifecycle (3 of 5)

- ▶ As state before, the container controls the servlet life cycle.
- ▶ The following methods are defined in the *GenericServlet* interface.
 - ▶ *init*: executed when the servlet is loaded.
 - ▶ *destroy*: executed when the servlet is finalized.
 - ▶ Some reasons: Container shutdown, new deployment
 - ▶ *service*: execute when the servlet is invoked.
- ▶ The *HttpServlet* class overrides the service method to invoke the corresponding HTTP methods (get, post, delete, trace, etc.)

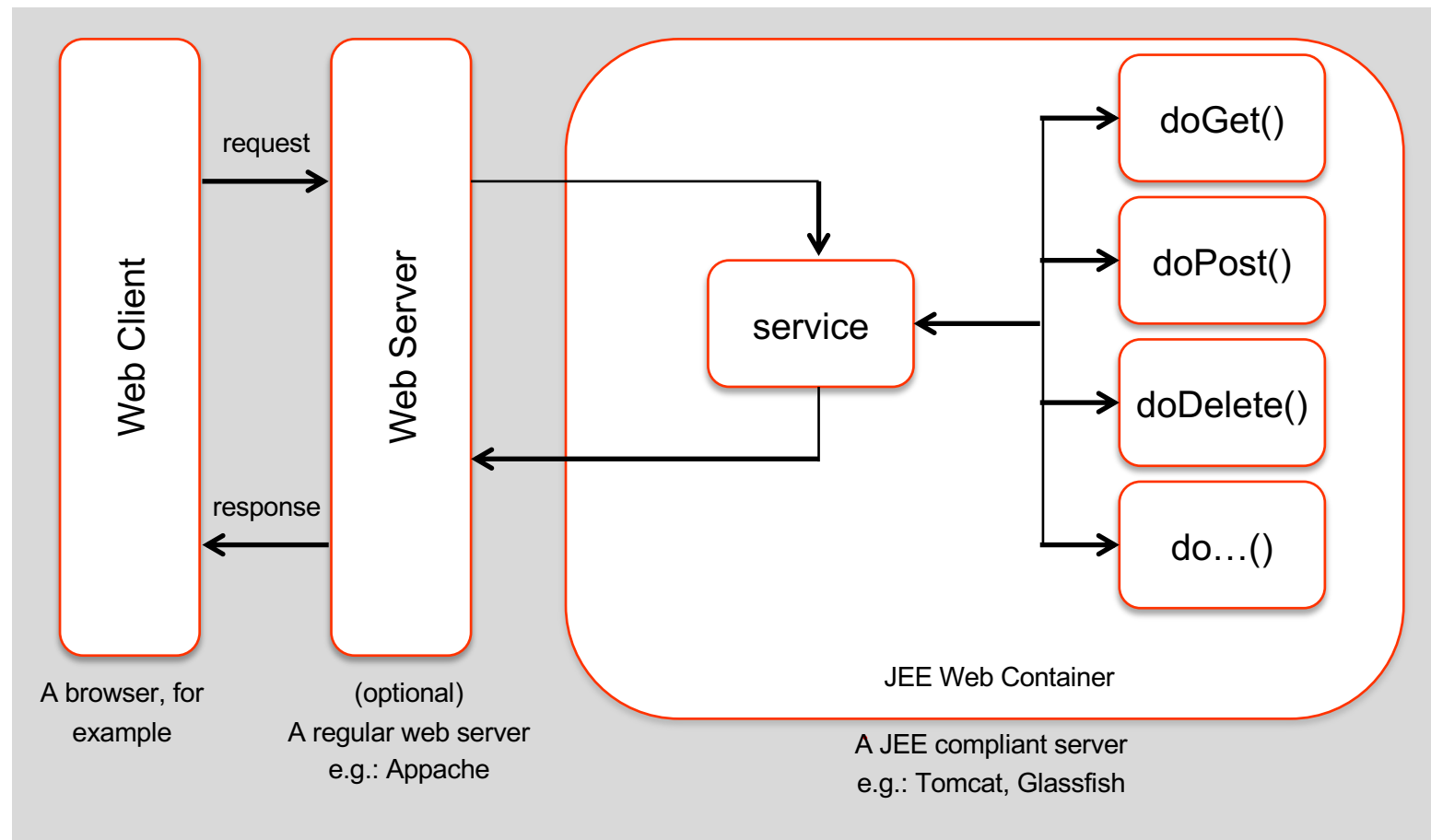


Servlet lifecycle (4 of 5)





Servlet lifecycle (5 of 5)





Generating responses (1 of 4)

- ▶ Textual responses can be written using a *PrintWriter* object:

```
PrintWriter out = response.getWriter ();
```

- ▶ `print()` and `println()` methods will write contents using the response stream

```
out.print ("JEE Course");
```

```
out.println ("Hello HvA");
```

- ▶ The content is sent to the client after flushing the output stream



Further reading

The Java EE 7 **Tutorial**: Eric Jendrock et al.

<http://docs.oracle.com/javaee/7/tutorial>

Java Servlet 4.0 **documentation**:

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

Java Servlet 4.0 Specification

<https://jcp.org/en/jsr/detail?id=369>