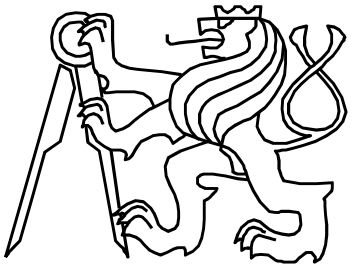
Enterprise Java (BI-EJA)
Technologie programování v jazyku Java (X36TJV)

Ing. Zdeněk Troníček, Ph.D.

Katedra softwarového inženýrství

Fakulta informačních technologií ČVUT v Praze



Letní semestr 2010/2011, přednáška č. 2
https://edux.fit.cvut.cz/courses/BI-EJA
https://edux.feld.cvut.cz/courses/X36TJV

Program

- Seznámení s Java Enterprise Edition (JEE)
- Aplikační server
- Architektura aplikace v JEE
- Servlety
- Java Server Pages (JSP)

Java Editions

Java Micro Edition

Java Standard Edition

Java Enterprise Edition

Motivace

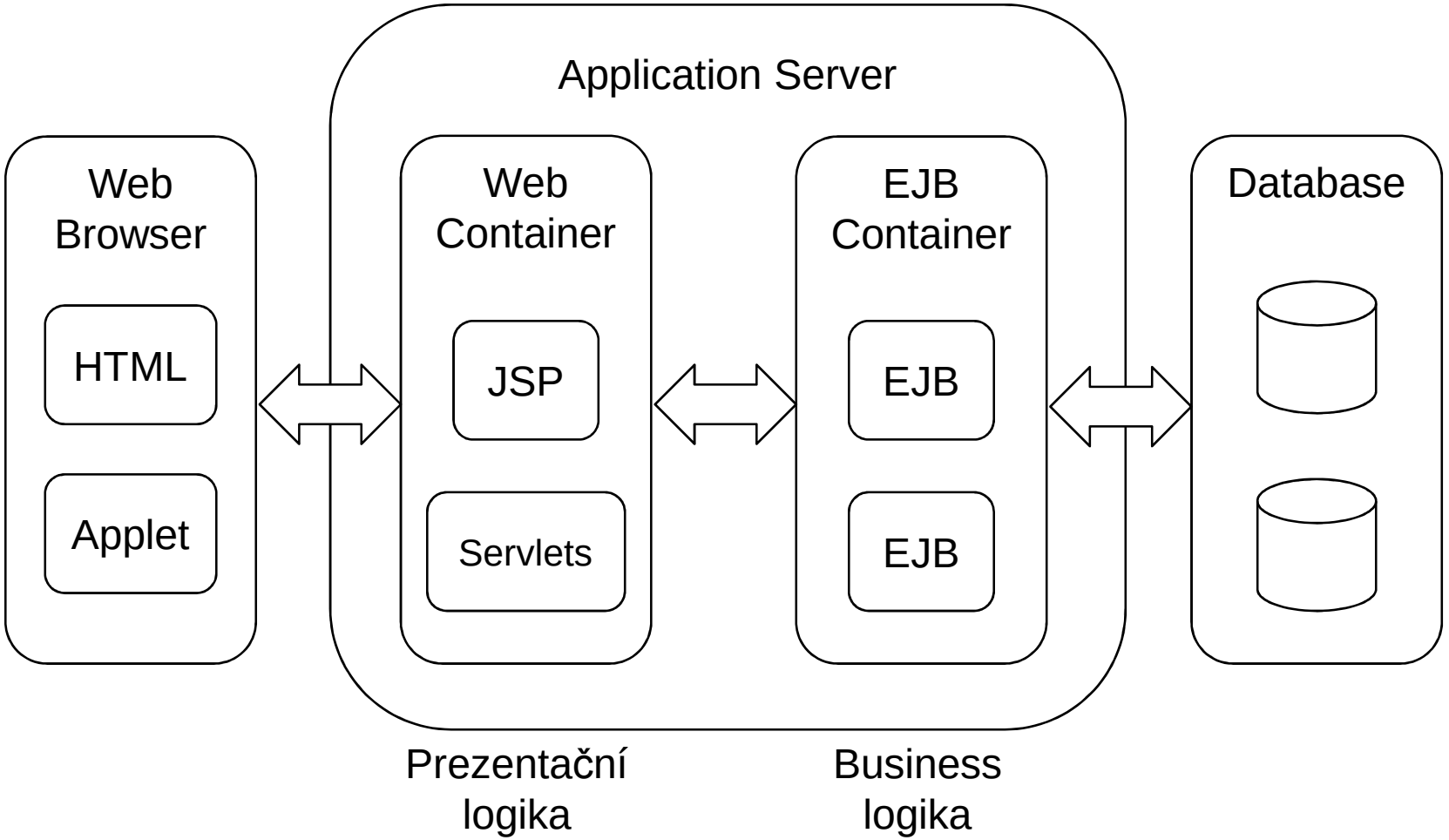
Co mají podnikové aplikace společného?

- User authentication
- Multi-user support
- Data persistence
- Data integrity (transactions)
- Client-tier communication
- Asynchronous communication
- Naming service
- Communication with legacy systems

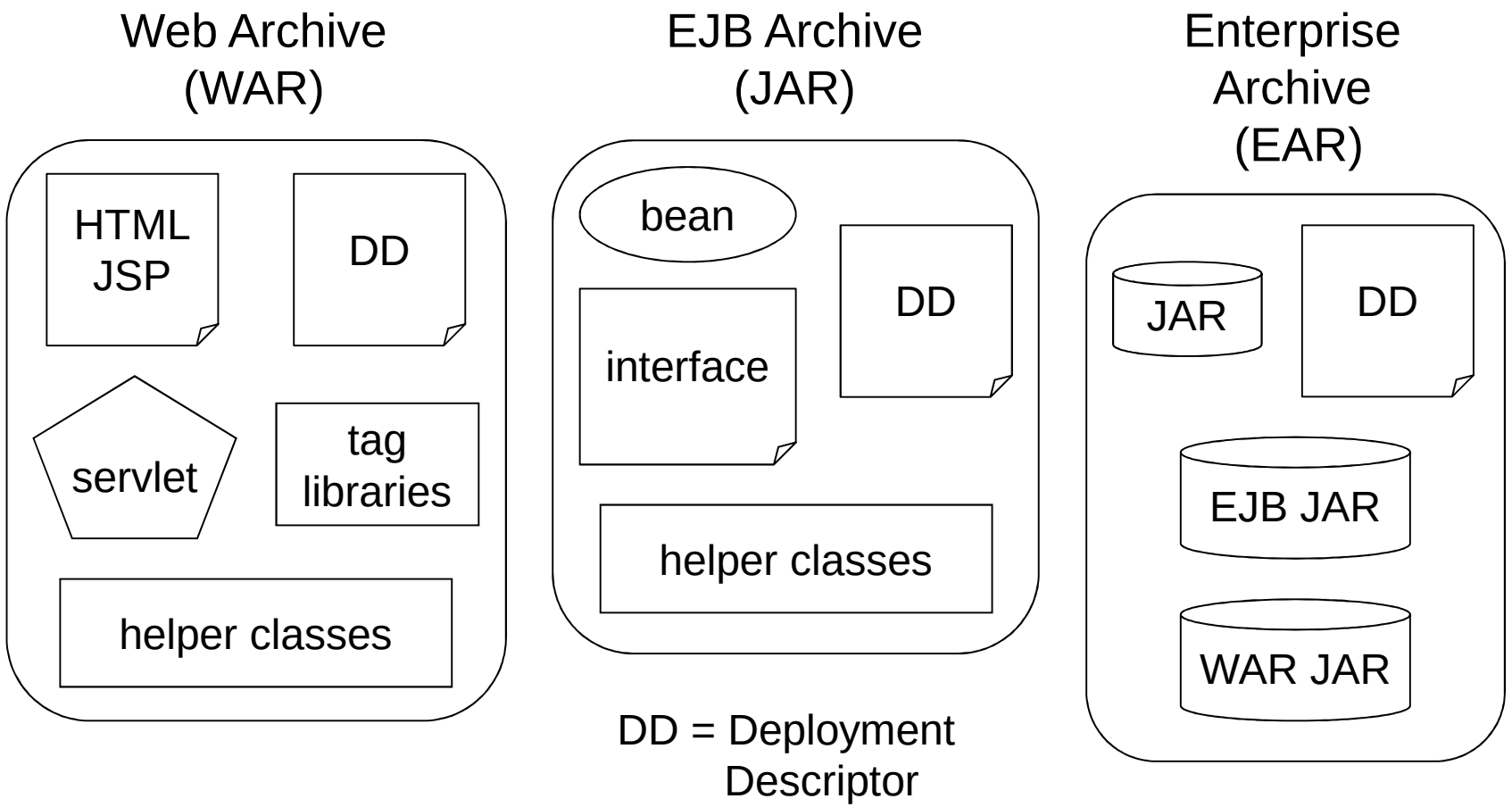
Java Enterprise Edition 6

- Java Server Faces (JSF) 2.0
- Java Server Pages (JSP) 2.2
- Enterprise Java Beans (EJB) 3.1
- Java Persistence API (JPA) 2.0
- Java API for XML-Based Web Services (JAX-WS) 2.2
- Java API for RESTful Web Services (JAX-RS) 1.1
- Java Architecture for XML Binding (JAXB) 2.2
- Java Message Service API (JMS) 1.1
- Java Transaction API (JTA) 1.1
- …

Příklad aplikace v JEE

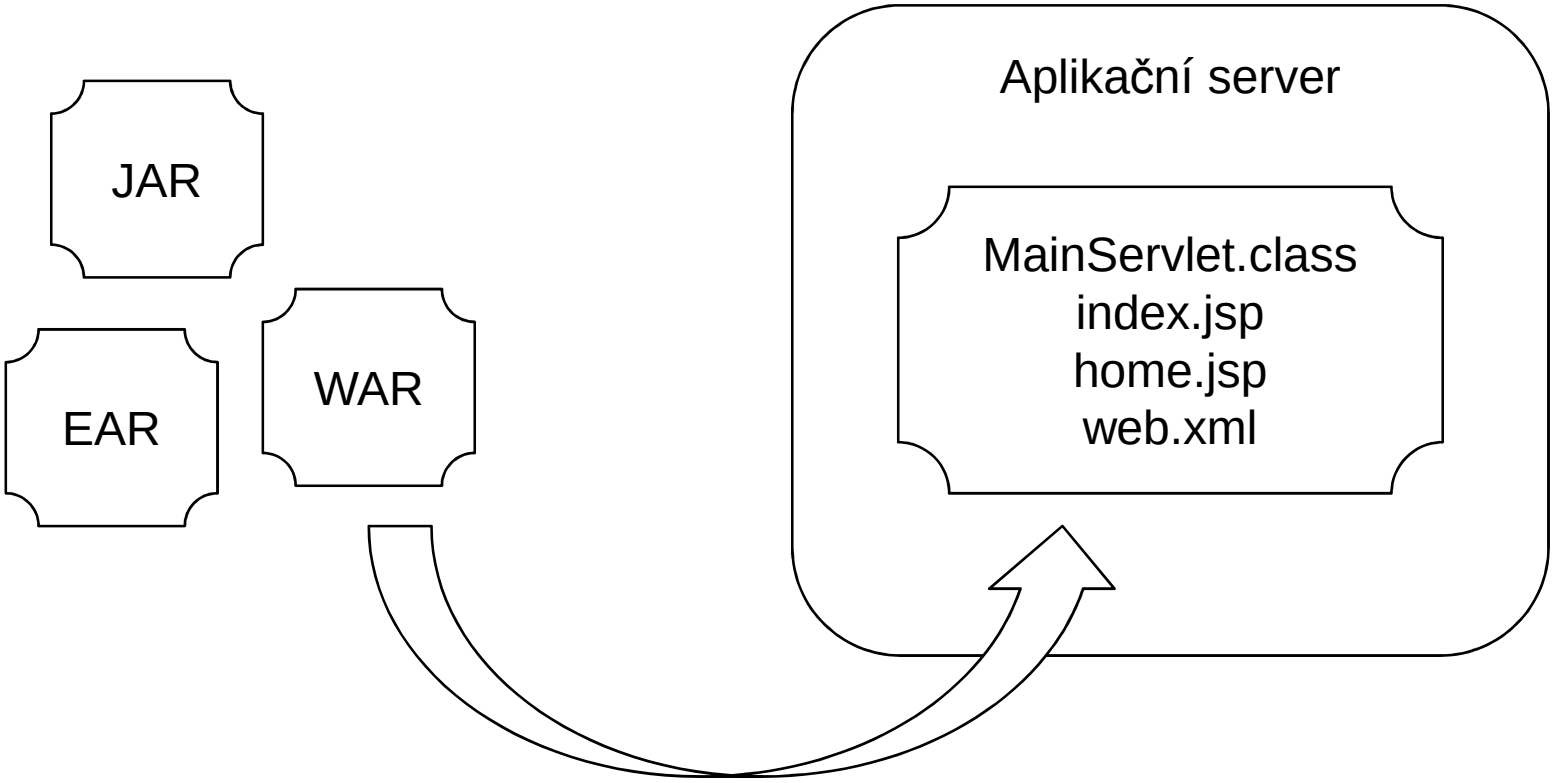


Archive files



DD = Deployment Descriptor

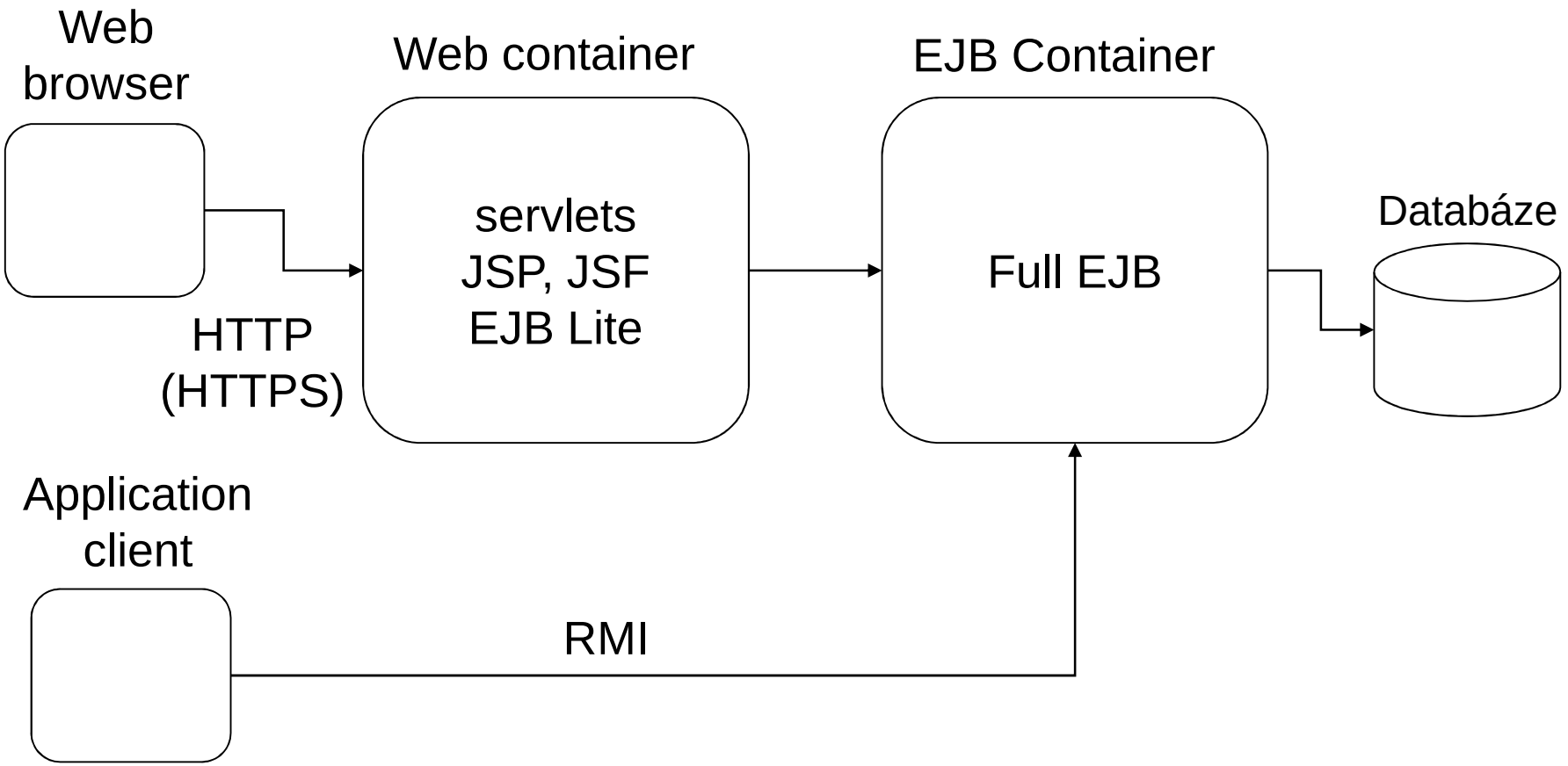
Deployment


JAR
EAR
WAR
Aplikační server
MainServlet.class
index.jsp
home.jsp
web.xml

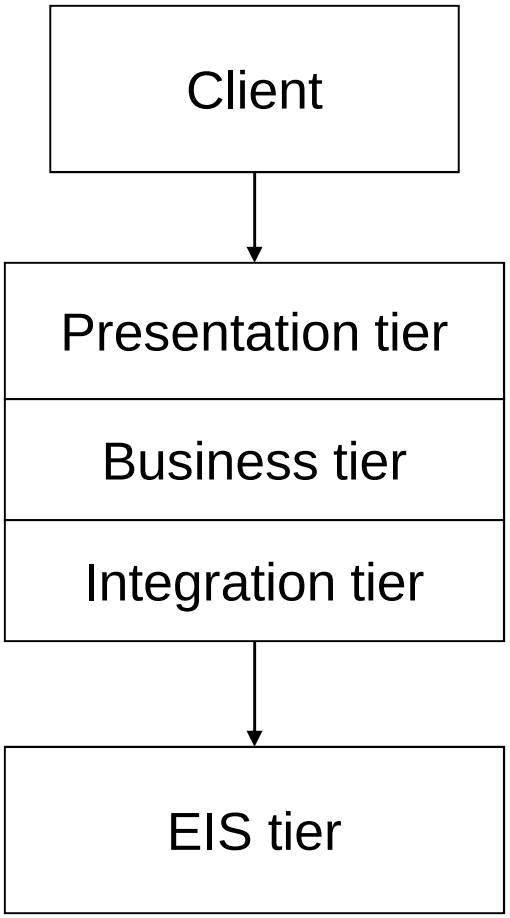
Služby poskytované AS

- Komunikace s klientem
- Životní cyklus komponent (zahrnuje i řízení souběžného přístupu)
- Správa databázových spojení (Connection pool)
- Transakční zpracování
- Persistence objektů
- Asynchronní komunikace
- Přihlašování uživatelů a přidělování práv
- …

Containers


Web
browser
Web container
EJB Container
servlets
JSP, JSF
EJB Lite
Full EJB
Databáze
HTTP
(HTTPS)
Application
client
RMI

N-tier architecture



web browser

servlets, JSP, HTML, XML

EJB, Plain Old Java Objects (POJO)

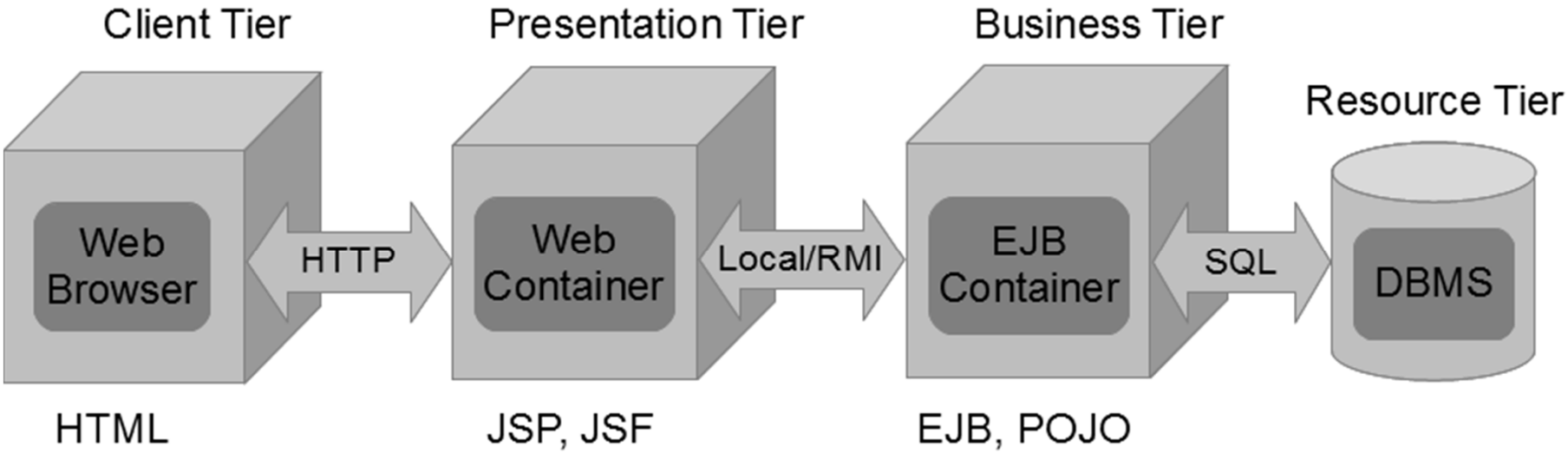
EJB, Entity, POJO

Relational Database Management System (RDBMS), Enterprise Resource Planning (ERP)

Archetypy

- Web application
- Rich Internet application
- Rich client application
- Mobile application
- Service application

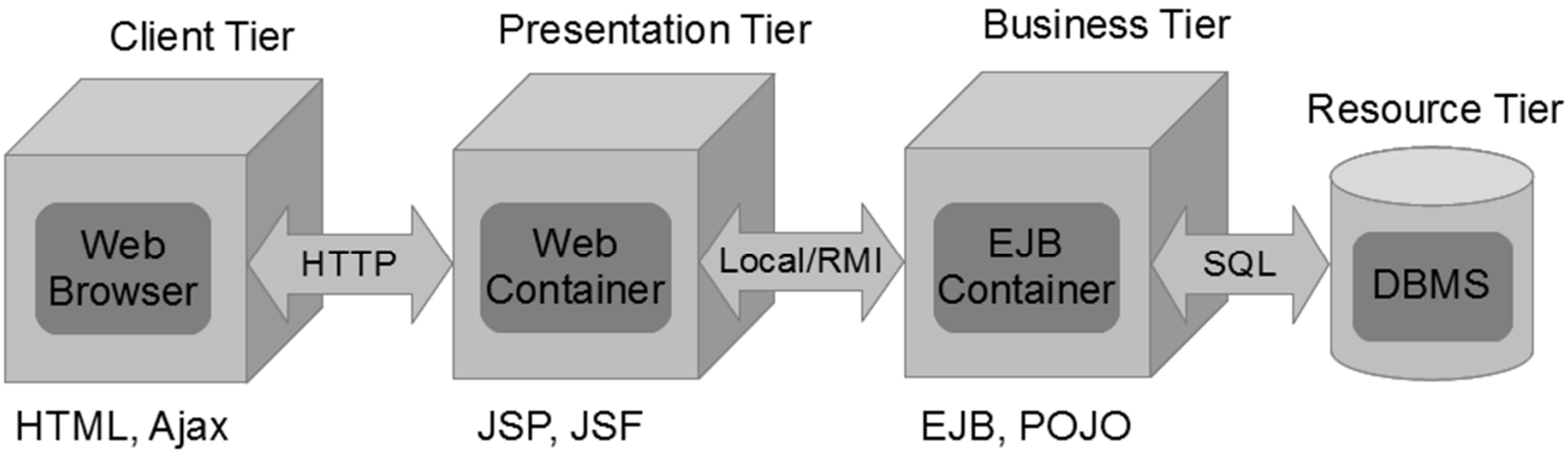
Web Application



+ modest client hardware
+ portability
+ simple upgrade
+ simple management

- simple UI
- connected scenarios only

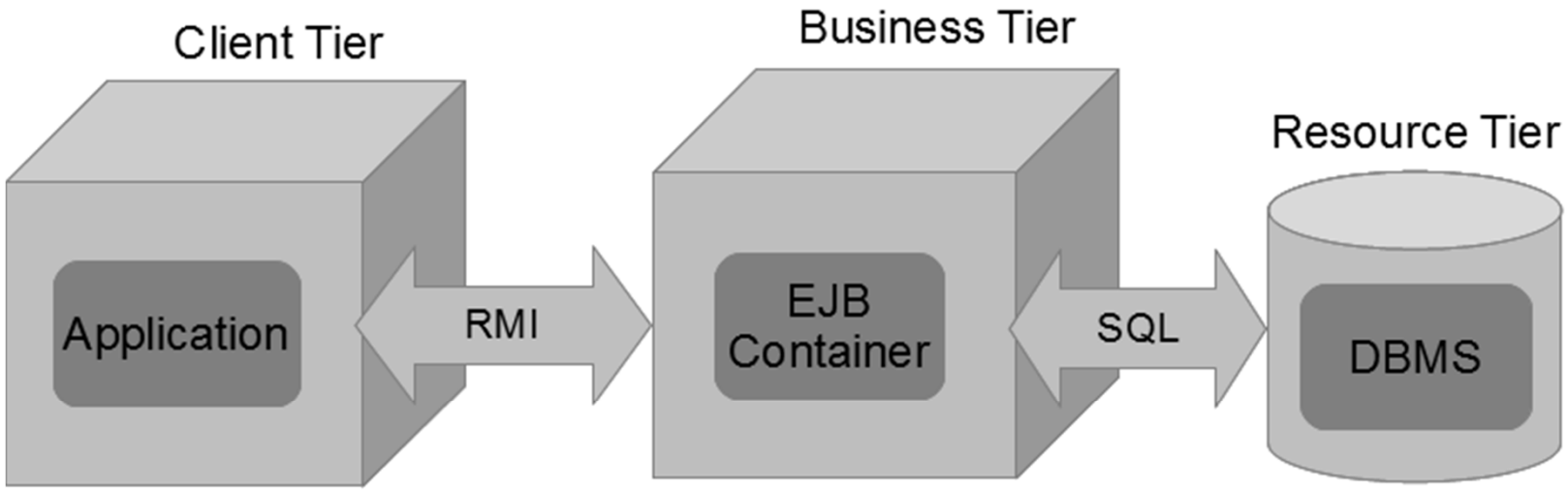
Rich Internet Application



+ rich UI
+ support for streaming media
+ simple upgrade
+ simple management

- requires runtime framework

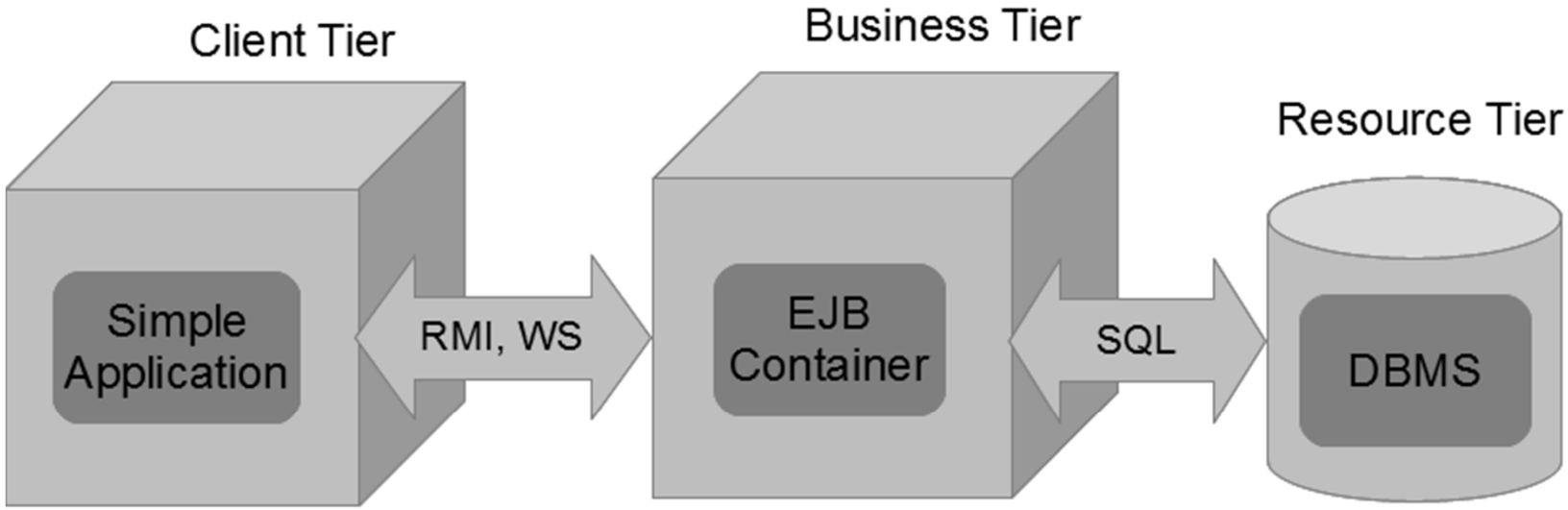
Rich Client Application



+ rich UI
+ interactive and responsive UI
+ offline support

- upgrade
- management

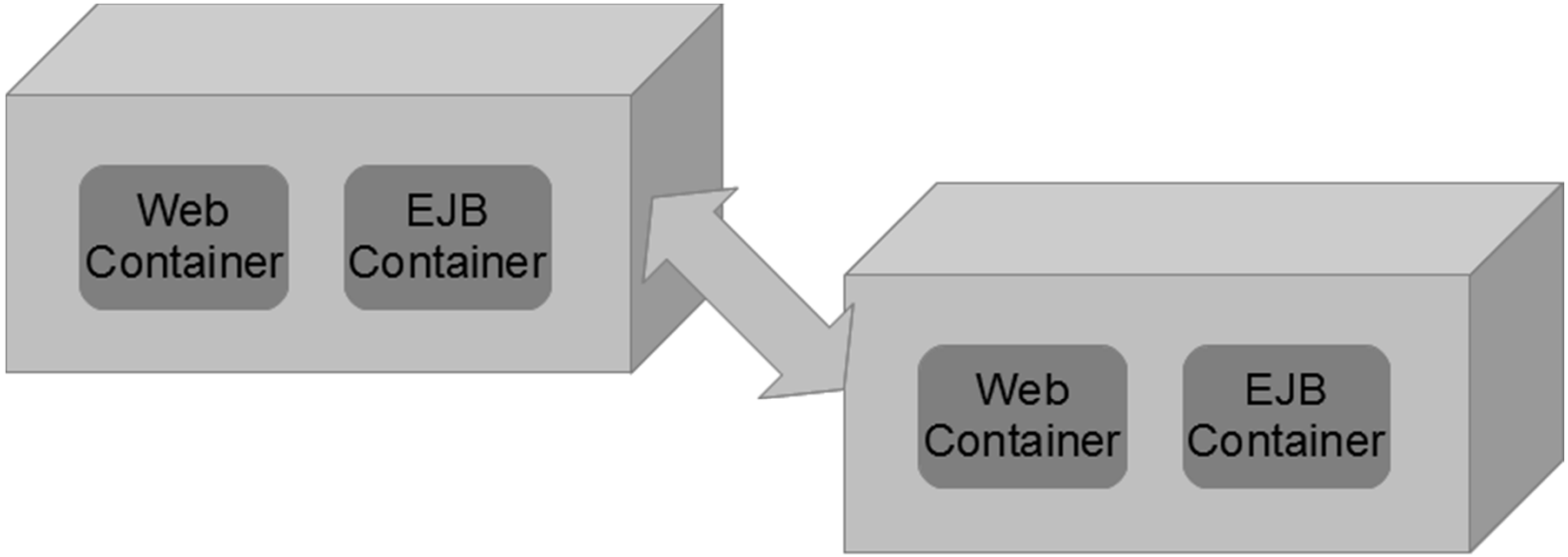
Mobile Application



+ support for handheld devices
+ offline support

- simple UI
- upgrade
- management

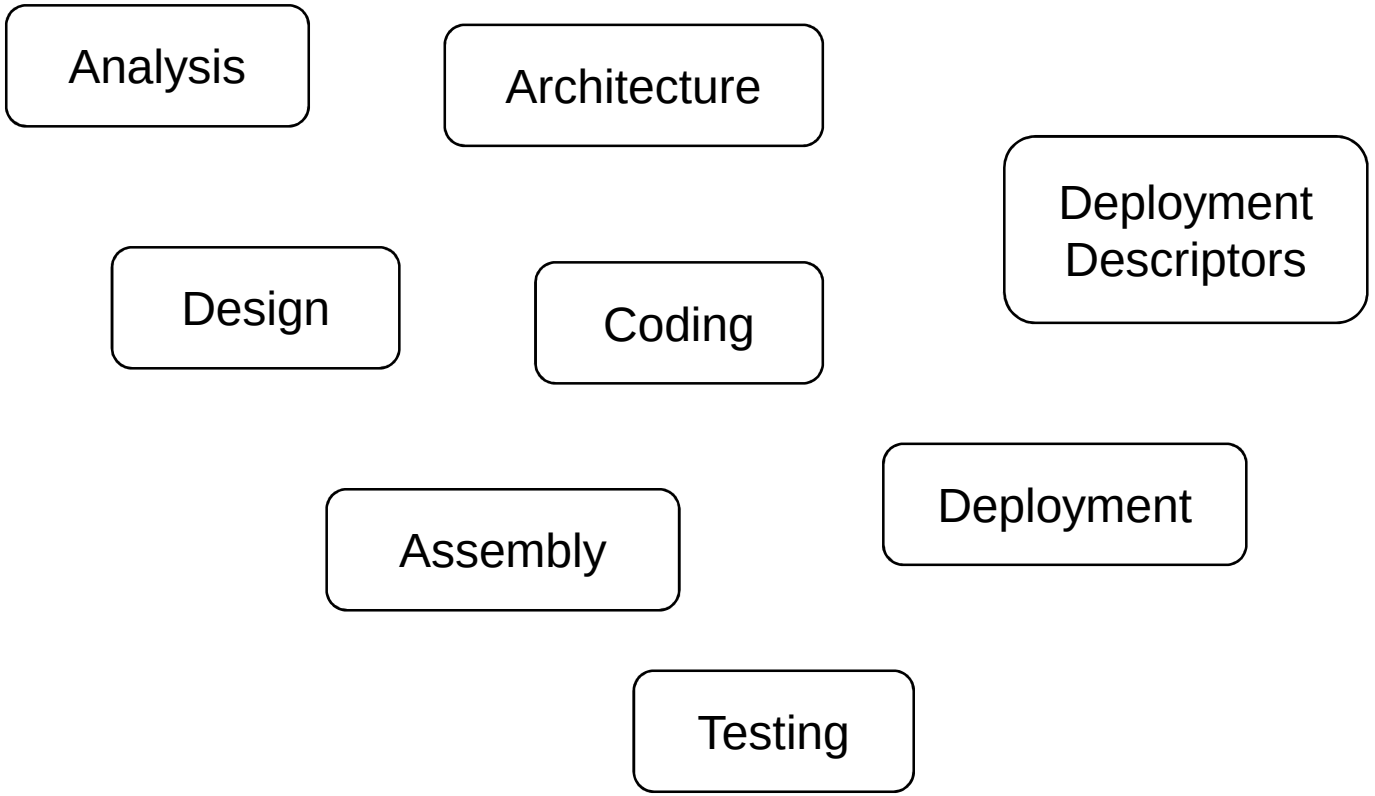
Service Application



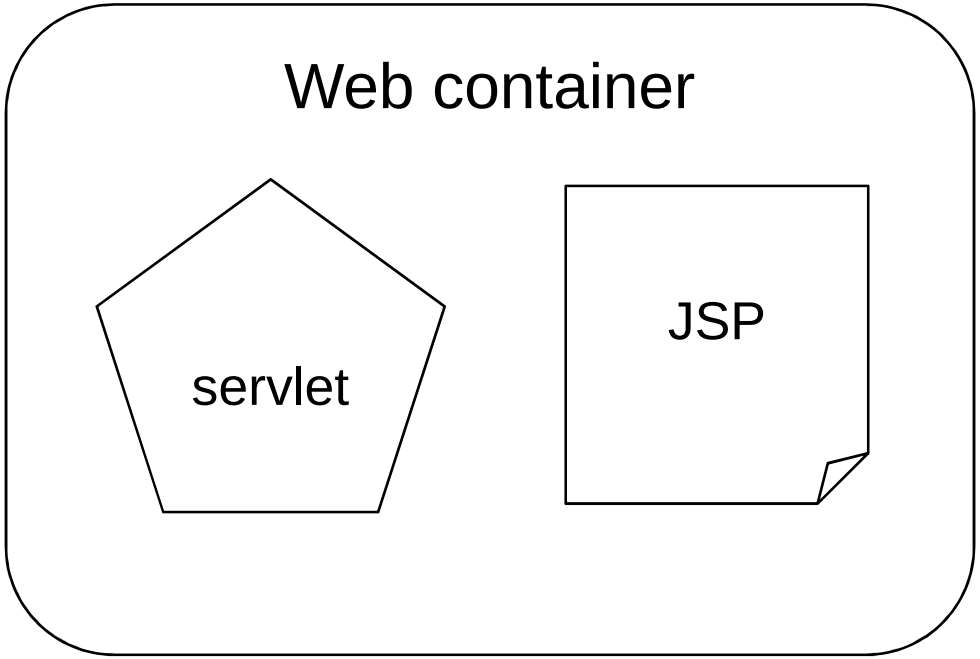
+ loose coupling
+ no UI

- XML processing

Development

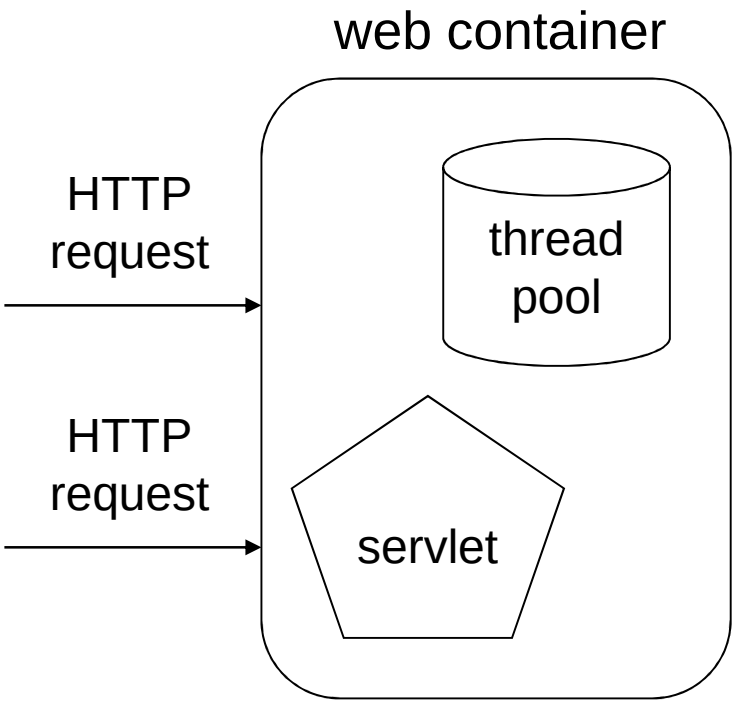


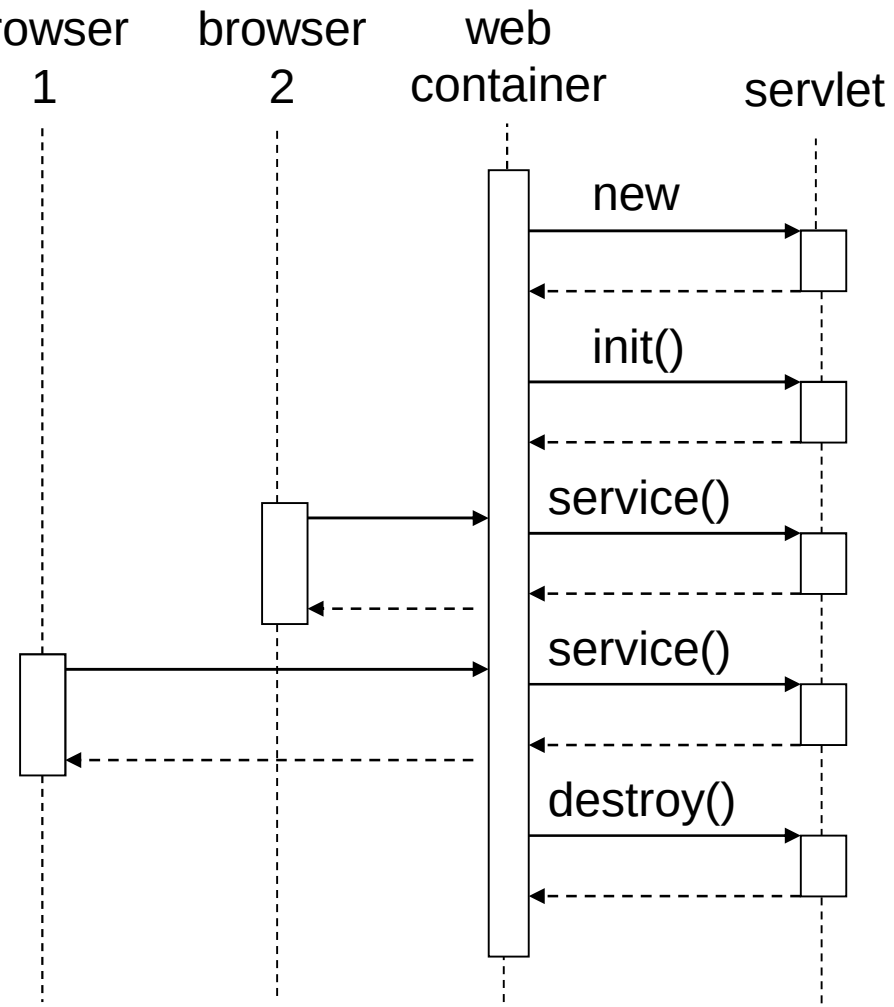
Servlety & JSP



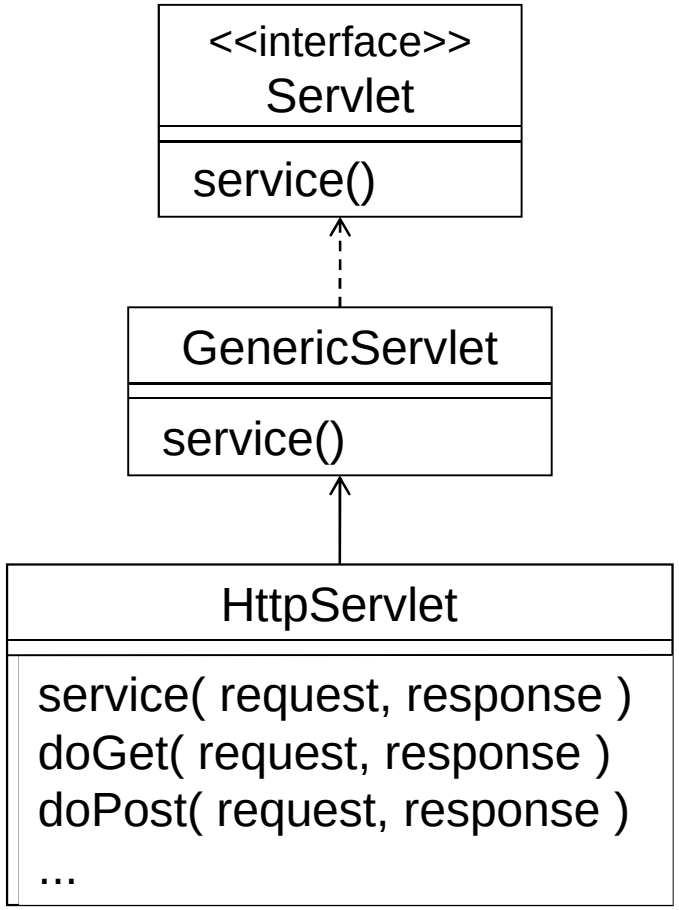
- Webové komponenty, které slouží pro dynamické generování stránek
- Pro daný servlet existuje v jednom kontejneru pouze jedna instance
- Každé JSP je před „provedením“ převedeno na servlet

Servlet


web container
HTTP request
HTTP request
thread pool
servlet


browser 1
browser 2
web container
servlet
new
init()
service()
service()
destroy()

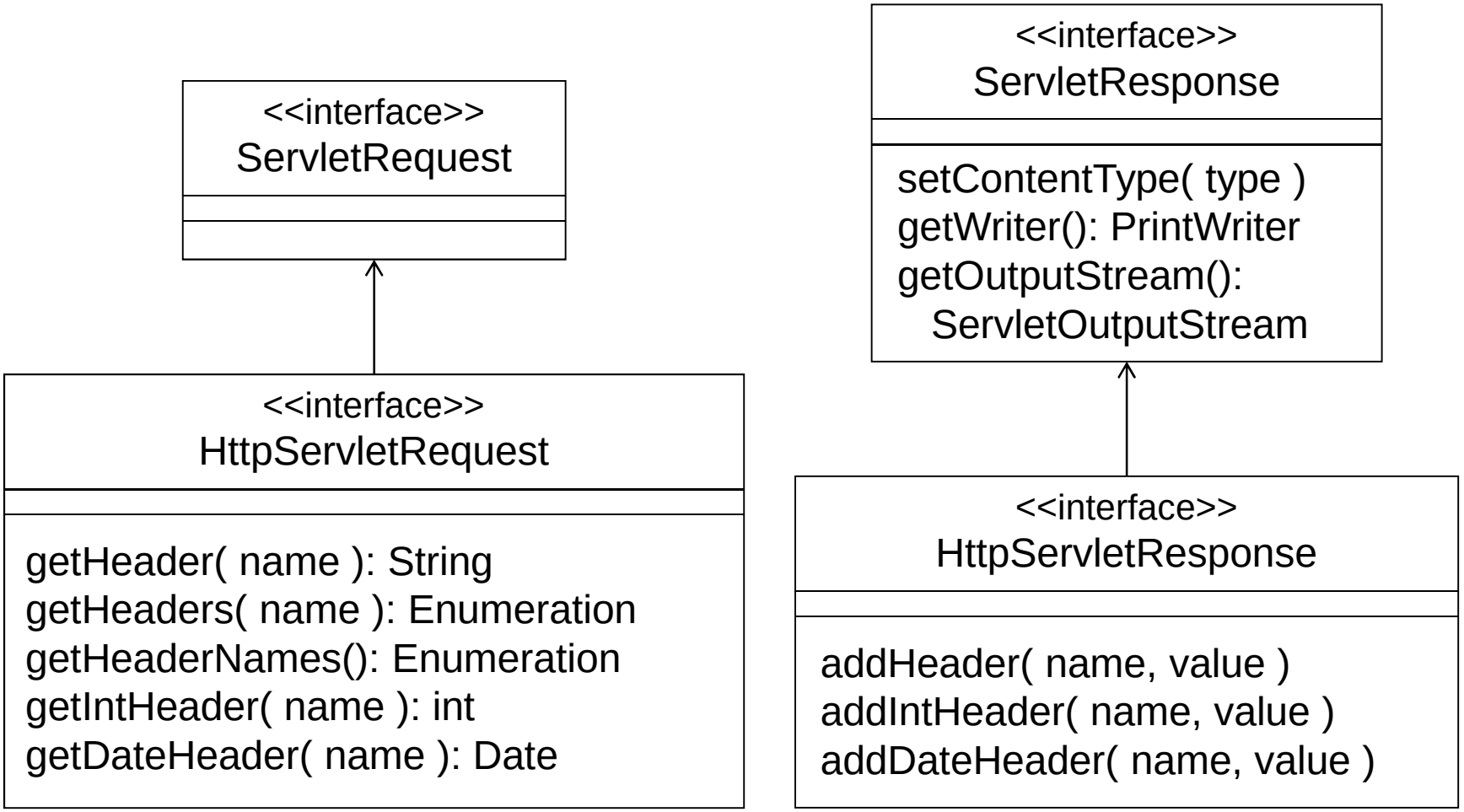
Servlet API



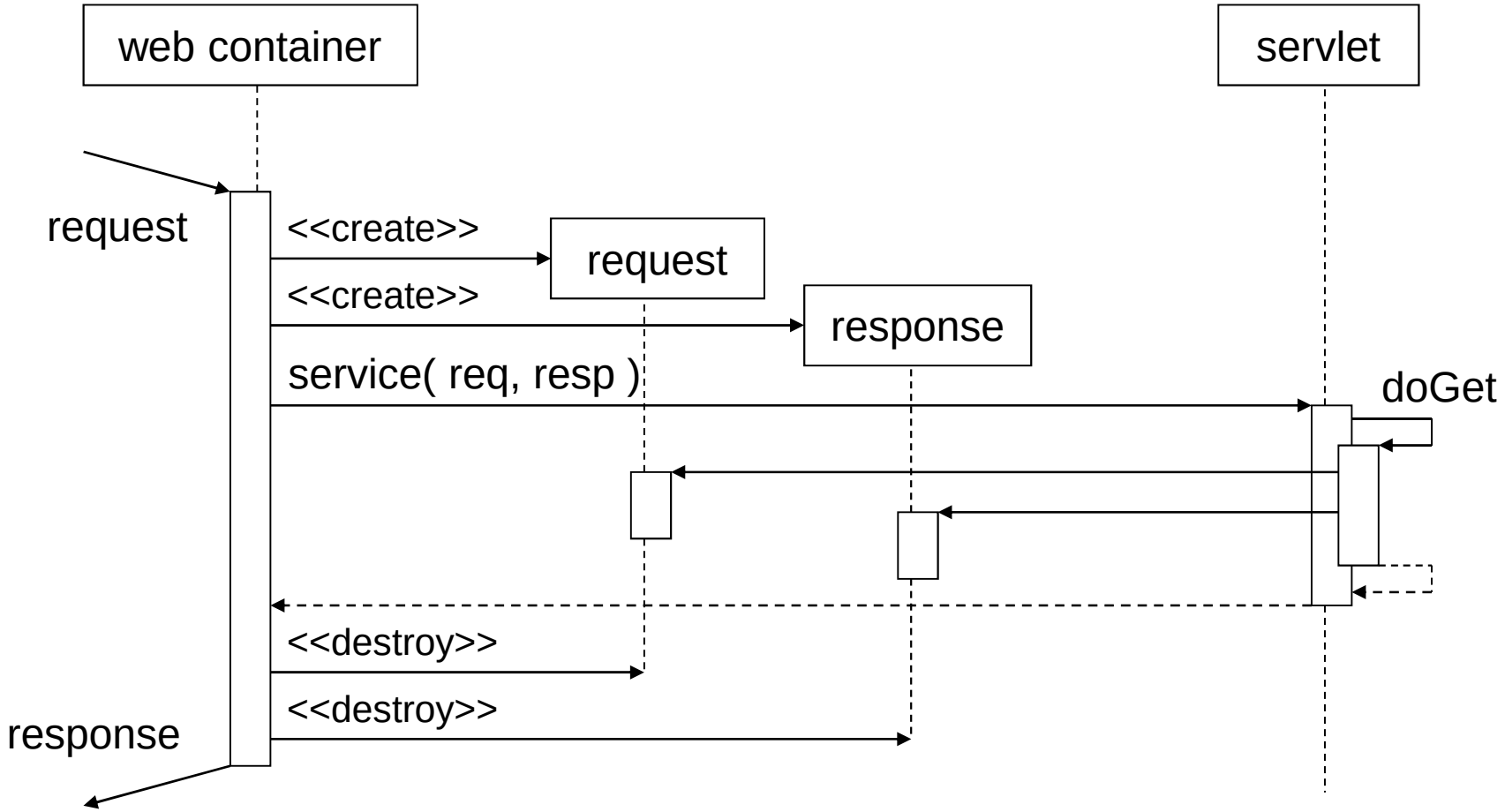

```
public class HelloServlet
    extends HttpServlet {

  @Override
  protected void doGet(
    HttpServletRequest req,
    HttpServletResponse resp )
    throws ... {
        resp.setContentType( "text/html" );
        PrintWriter out = resp.getWriter();
        ...
  }
}
```

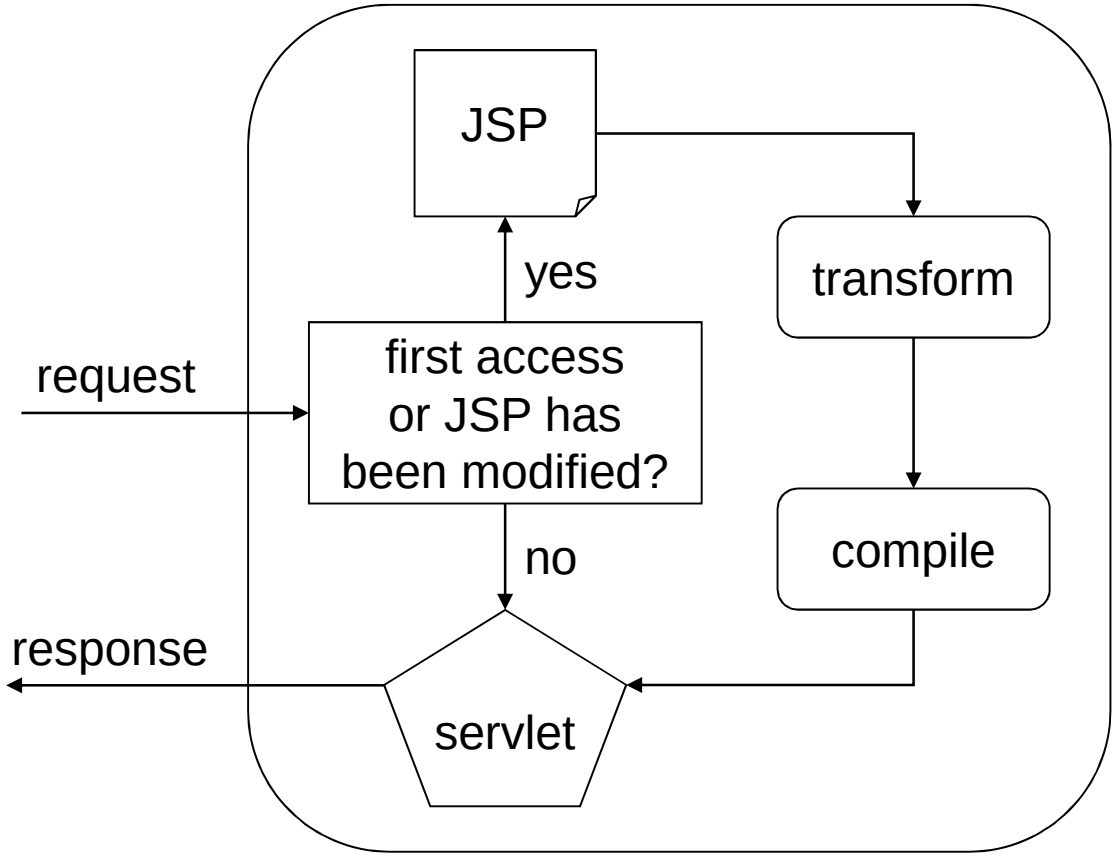
Request & Response API



Zpracování požadavku



Java Server Pages (JSP)




```
<%@ page
  import="java.util.Date" %>
<html>
  <body>
    <h1>EJA</h1>
    <%= new Date() %>
  </body>
</html>
```

JSP scripting elements

Directives

```
<%@ page import="..."%>
<%@ include file="..."%>
<%@ taglib uri="..." prefix="..."%>
```

Declarations

```
<%! String url = "/account.jsp"%>
<%!
  boolean numOfAccounts(Customer c) {
    return c.getAccounts().size();
  } %>
```

Expressions

```
<%= c.getAccounts().size() %>
```

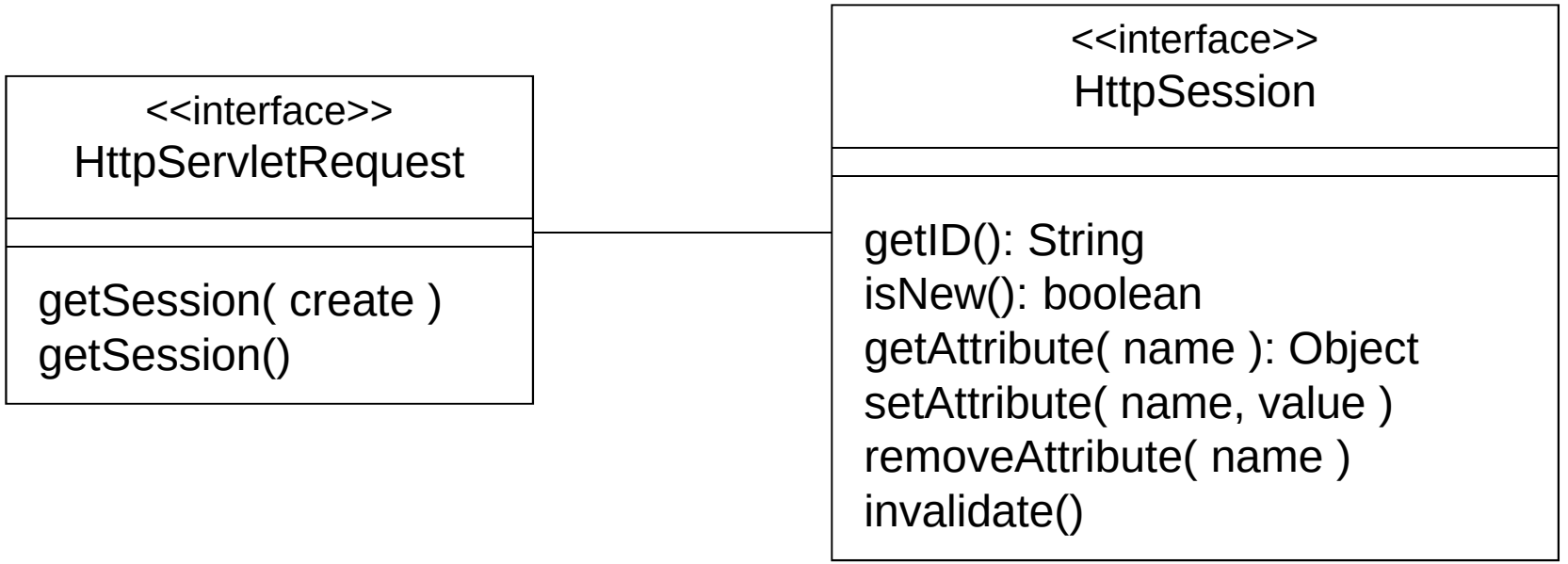
Scriptlets

```
<%
  out.println("Hi!");
%>
```

Comments

```
<%-- not finished yet! --%>
```

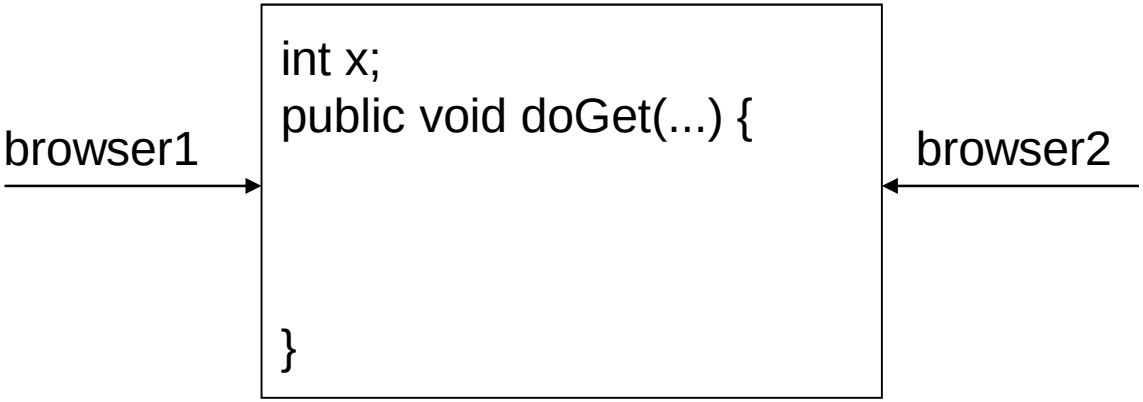
Session API



web.xml

```
<session-config>
    <session-timeout>10</session-timeout>
</session-config>
```

Thread model



instance variables
class variables
external resources
synchronized keyword

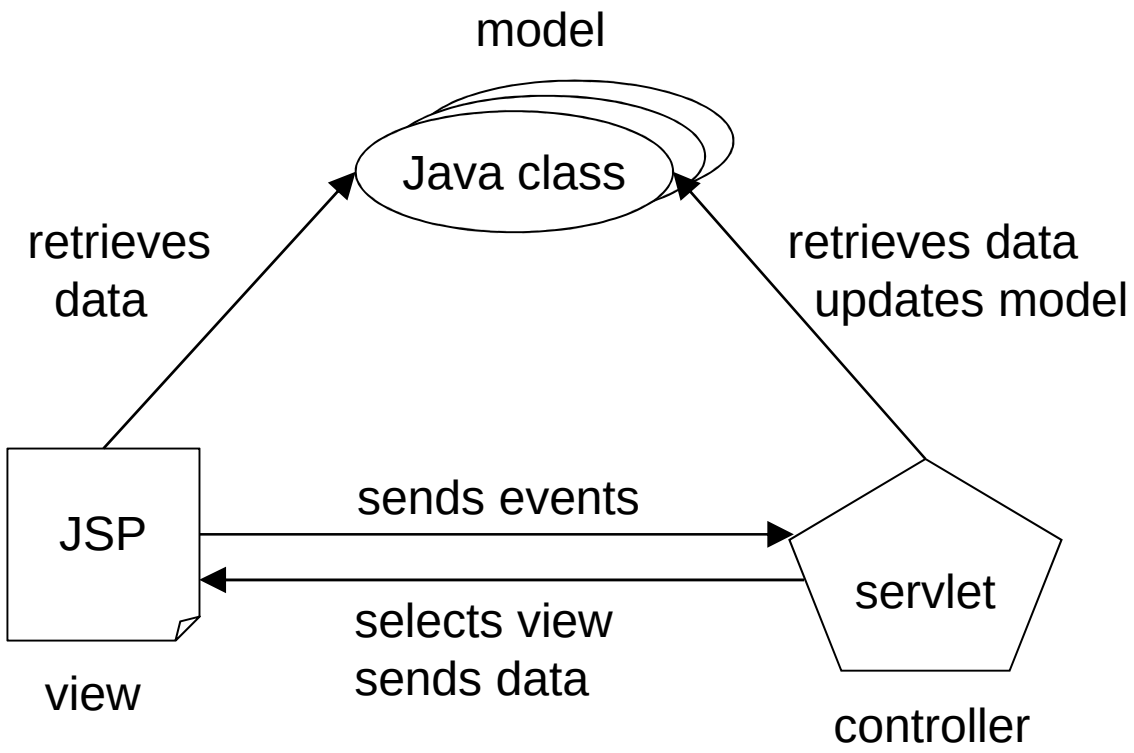
```
Lock lock = new ReentrantLock();
lock.lock();
try {
    ...
} finally {
    lock.unlock();
}
```

Thread interference

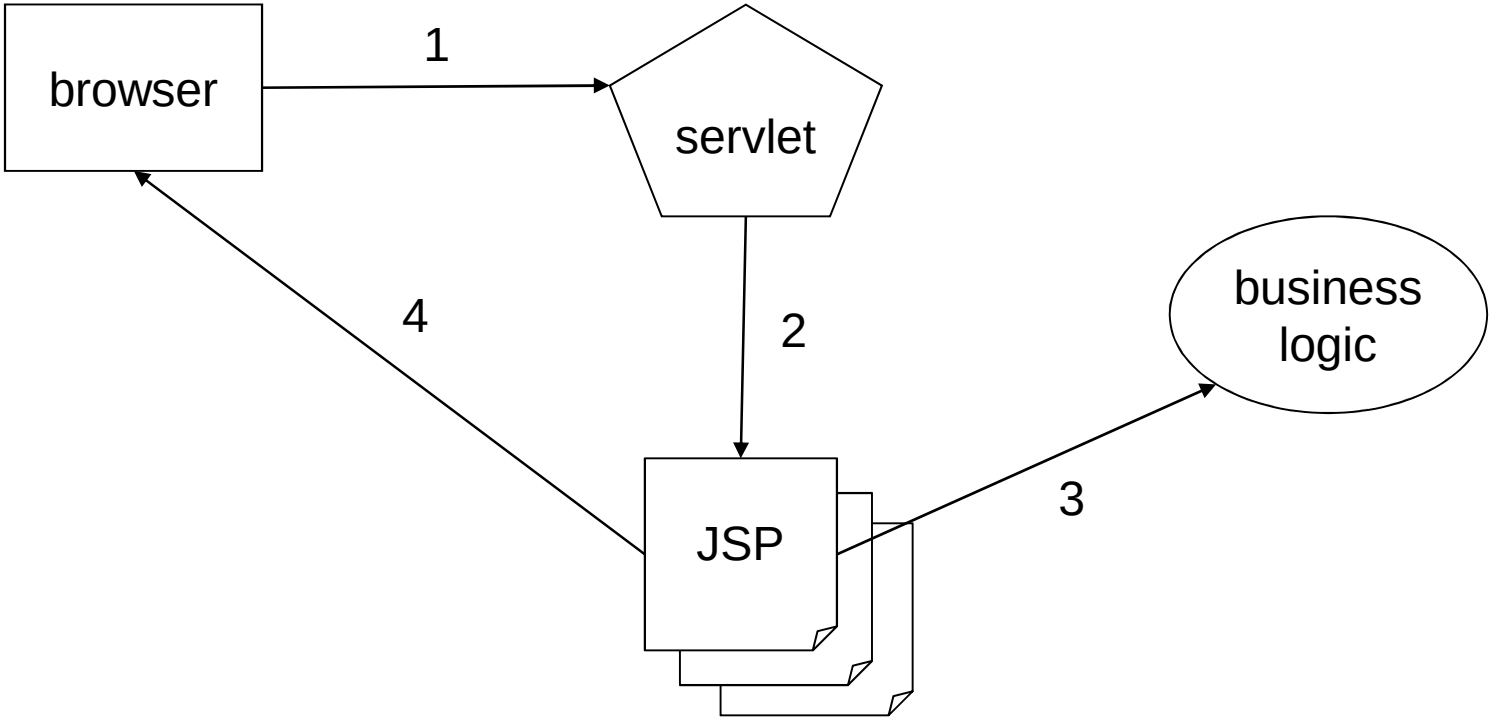
```
class Counter {
    private int c = 0;
    public void increment() { c++; }
    public void decrement() { c--; }
    public int value() { return c; }
}
```

Vlákno A volá increment(), vlákno B decrement():
1. vlákno A přečte c
2. vlákno B přečte c
3. vlákno A zvýší c o 1
4. vlákno B sníží c o 1
5. vlákno A uloží výsledek (v c je 1)
6. vlákno B uloží výsledek (v c je -1)

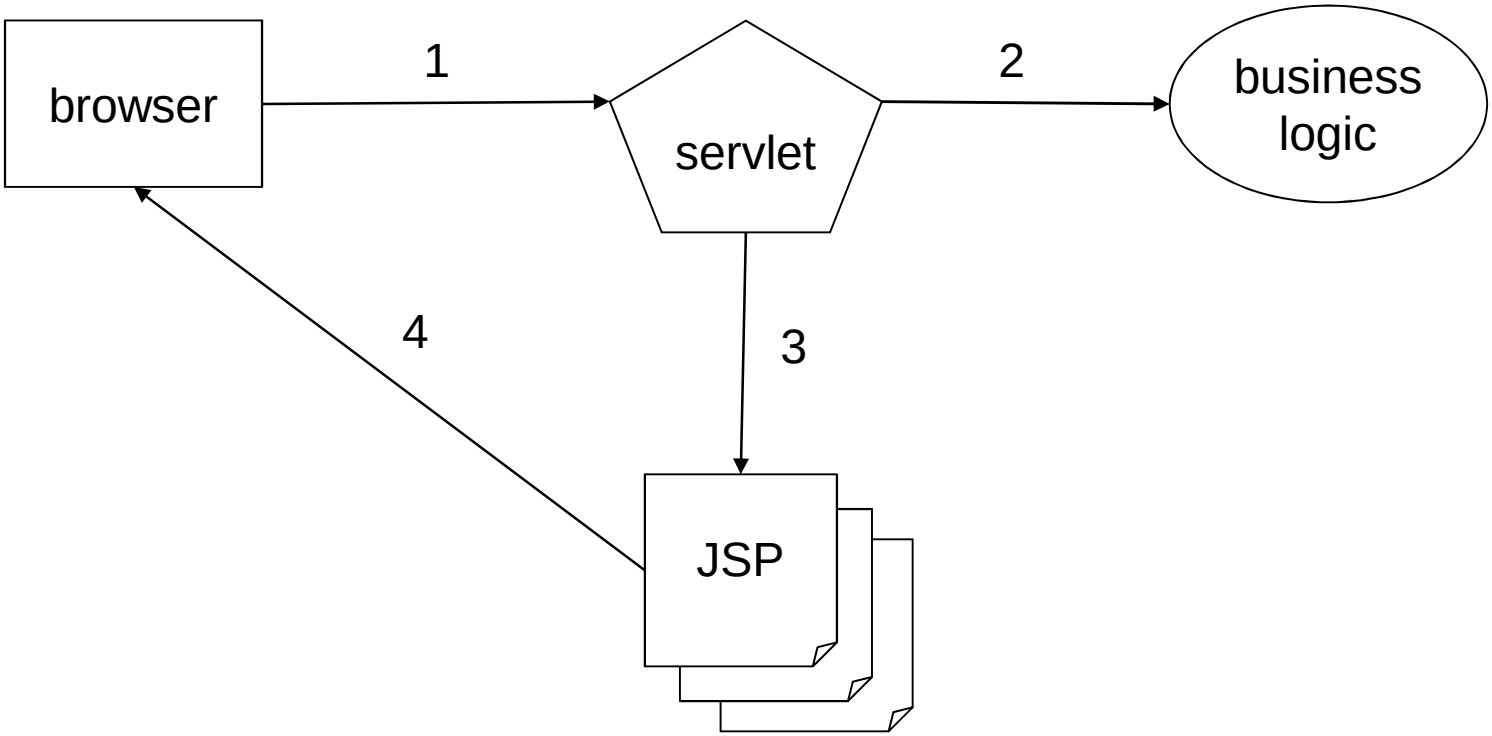
Model 2 (Model-View-Controller)



Dispatcher View



Service-to-Worker



Předávání dat

Controller

```
Customer c = ...
request.setAttribute( "customer", c );
RequestDispatcher rd =
    request.getRequestDispatcher( "/view.jsp" );
rd.forward( request, response );
```

Scope

page
request
session
application

View

```
<jsp:useBean id="customer" class="bank.Customer" scope="request"/>
<jsp:getProperty name="customer" property="firstName"/>
<jsp:getProperty name="customer" property="lastName"/>
```

Rozdělení rolí

servlet	JSP
• přijme žádost • zpracuje formulářová data • připraví data pro view • předá řízení JSP	• přijme data od servletu • vytvoří HTML odpověď

generic servlet

Java Server Faces
Struts

Otázky & odpovědi

Znáte NetBeans API a chcete pracovat na zajímavém projektu? Napište mi!