Java Server Pages

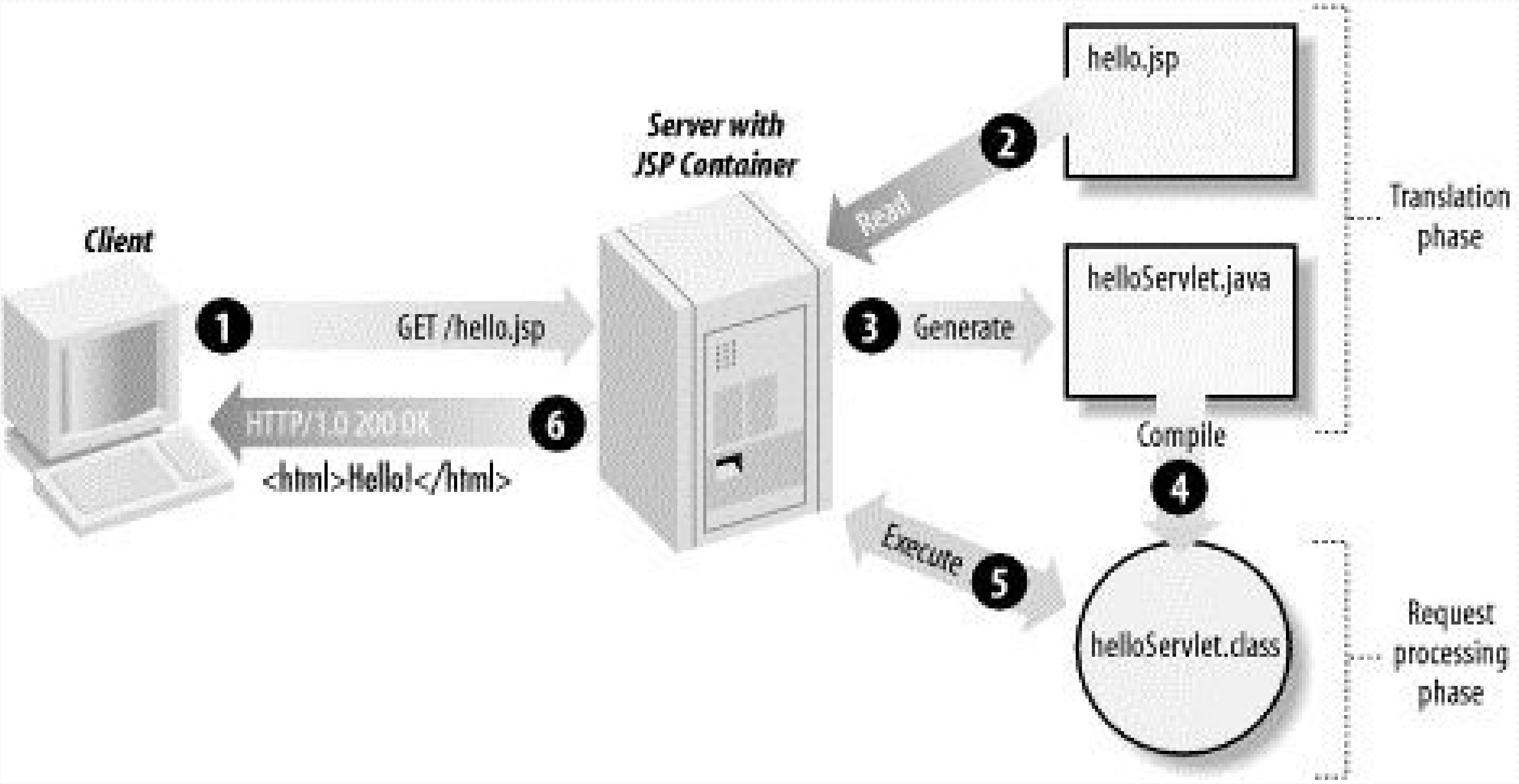
What is JSP?

- A JSP page consists of HTML tags and JSP tags..
- we can separate designing and development.
- Insert java code in HTML pages by making use of special JSP tags, most of which start with <% and end with %>.
- JSP can play a part in the simplest applications to the most complex and demanding.

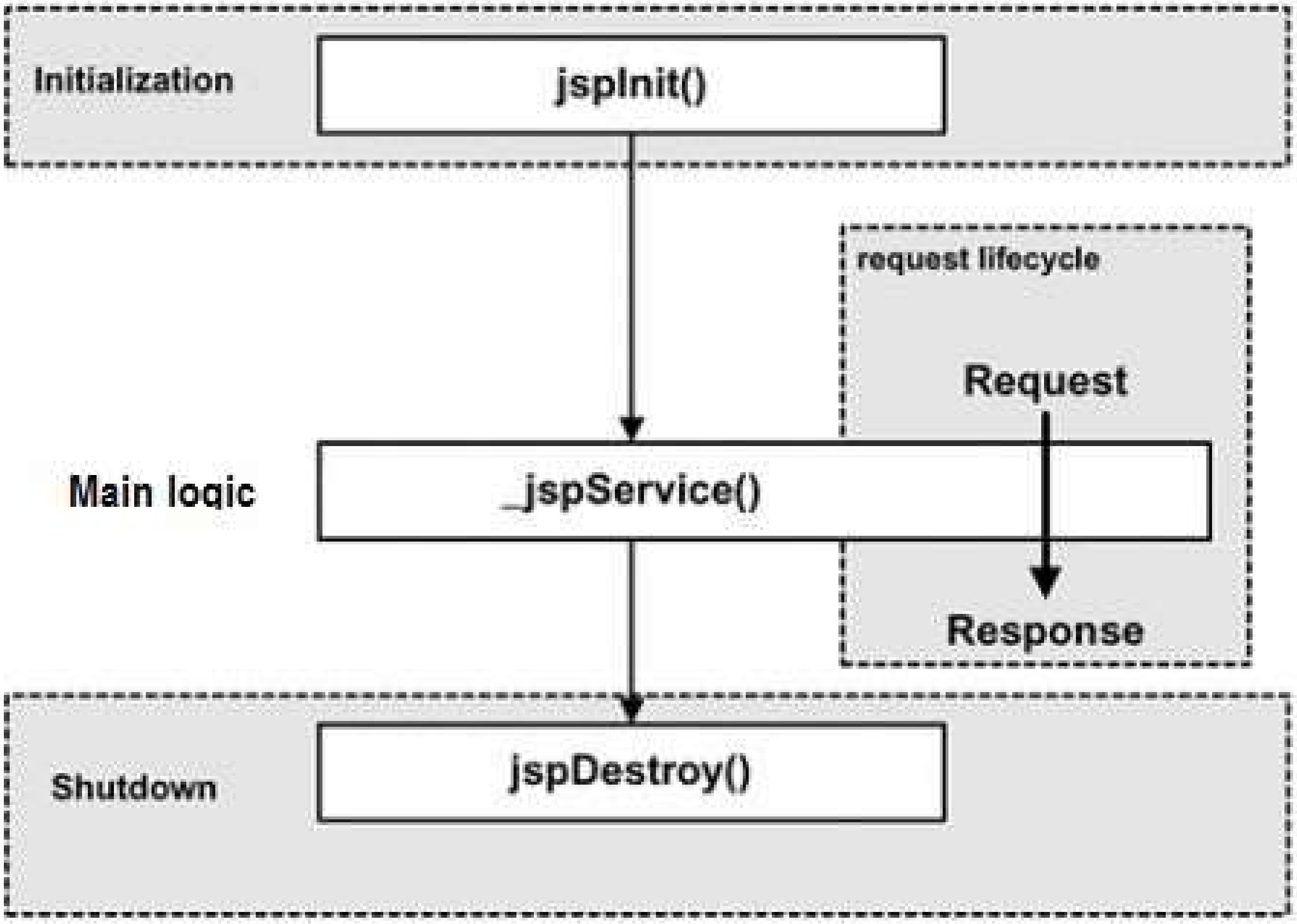
JSP vs Servlet

- Extension to Servlet
- Easy to maintain
- Fast Development: No need to recompile and redeploy
- Less code than Servlet
- It is more convenient to write regular HTML than to have plenty of println statements that generate the HTML.
- Separation of role as developer & designer

JSP Life Cycle



JSP Life Cycle methods



JSP Scriptlet tag

- Scripting elements
- The scripting elements provides the ability to insert java code inside the jsp.
- There are three types of scripting elements:
  - scriptlet tag
  - expression tag
  - declaration tag
- <% code fragment %>
- <% out.print("welcome to jsp"); %>

JSP Declarations

- A declaration declares one or more variables or methods that you can use in Java code later in the JSP file.
- You must declare the variable or method before you use it in the JSP file.
- <%! declaration; [declaration;]+ ... %>
- Example
  - <%! int i = 0; %>
  - <%! int a, b, c; %>
  - <%! Circle a = new Circle(2.0); %>

JSP Expression

- The code placed within expression tag is written to the output stream of the response.
- You need not write out.print() to write data.
- <%= statement %>
- Example

```
<p>
   Today's date: <%= (new
   java.util.Date()).toLocaleString()%>
</p>
```

JSP Comments

- <%-- This is JSP comment --%>

Syntax	Purpose
<%-- comment --%>	A JSP comment. Ignored by the JSP engine.
<!-- comment -->	An HTML comment. Ignored by the browser.
<\%	Represents static <% literal.
%\>	Represents static %> literal.
\'	A single quote in an attribute that uses single quotes.
\"	A double quote in an attribute that uses double quotes.

JSP Implicit Objects

Objects	Description
request	This is the **HttpServletRequest** object associated with the request.
response	This is the **HttpServletResponse** object associated with the response to the client.
out	This is the **PrintWriter** object used to send output to the client.
session	This is the **HttpSession** object associated with the request.
application	This is the **ServletContext** object associated with application context.
config	This is the **ServletConfig** object associated with the page.
pageContext	This encapsulates use of server-specific features like higher performance**JspWriters**.
page	This is simply a synonym for **this**, and is used to call the methods defined by the translated servlet class.
Exception	The **Exception** object allows the exception data to be accessed by designated JSP.

JSP - Directives

- JSP directives provide directions and instructions to the container

Directive	Description
<%@ page ... %>	Defines page-dependent attributes, such as scripting language, error page, and buffering requirements.
<%@ include ... %>	Includes a file during the translation phase.
<%@ taglib ... %>	Declares a tag library, containing custom actions, used in the page

JSP Action Tags

Syntax	Purpose
jsp:include	Includes a file at the time the page is requested
jsp:useBean	Finds or instantiates a JavaBean
jsp:setProperty	Sets the property of a JavaBean
jsp:getProperty	Inserts the property of a JavaBean into the output
jsp:forward	Forwards the requester to a new page
jsp:plugin	Generates browser-specific code that makes an OBJECT or EMBED tag for the Java plugin
jsp:element	Defines XML elements dynamically.
jsp:attribute	Defines dynamically defined XML element's attribute.
jsp:body	Defines dynamically defined XML element's body.
jsp:text	Use to write template text in JSP pages and documents.

Jsp:forward

- The jsp:forward action tag is used to forward the request to another resource.
- Forward without parameter

```
<jsp:forward page="printdate.jsp" />
```

- Forward with parameter

```
<jsp:forward page="printdate.jsp" >
<jsp:param name="name" value="javatpoint.com" />
</jsp:forward>
```

Java Bean

- It is a reusable software component
- A Java Bean is a java class that should follow following conventions:
  - It should have a no-arg constructor.
  - It should be Serializable.
  - It should provide methods to set and get the values of the properties, known as getter and setter methods.

jsp:useBean

- It first searches for an existing object utilizing the id and scope variables.
- If an object is not found, it then tries to create the specified object.
- <jsp:useBean id="name" class="package.class" />

Jsp:useBean

- id: is used to identify the bean in the specified scope.
- scope: represents the scope of the bean. Values: page, request, session or application. default :page.
- class: instantiates the specified bean class
  - it must have no-arg or no constructor
  - must not be abstract.
- type: provides the bean a data type if the bean already exists in the scope.
  - If you use it without class or beanName, no bean is instantiated.
- beanName: instantiates the bean using the java.beans.Beans.instantiate() method.

jsp:setProperty

- The setProperty and getProperty action tags are used for developing web application with Java Bean.
- The jsp:setProperty action tag sets a property value or values in a bean using the setter method.
- set all the values of incoming request in the bean
  - <jsp:setProperty name="bean" property="*" />
- set value of the incoming specific property
  - <jsp:setProperty name="bean" property="username" />
- set a specific value in the property
  - <jsp:setProperty name="bean" property="username" value="Kumar" />

jsp:getProperty

- The jsp:getProperty action tag returns the value of the property.
- <jsp:getProperty name="obj" property="name" />